NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ASSESSMENT OF AN ONBOARD EO SENSOR TO ENABLE DETECT-AND-SENSE CAPABILITY FOR UAVs OPERATING IN A CLUTTERED ENVIRONMENT

by

Wee Kiong Ang

September 2017

Thesis Advisor:	Oleg Yakimenko
Co-Advisor:	Dong Hye Ye
Second Reader:	Fotis Papoulias

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704–0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 2017	Master's thesis

4. TITLE AND SUBTITLE
ASSESSMENT OF AN ONBOARD EO SENSOR TO ENABLE DETECT-AND-SENSE CAPABILITY FOR UAVs OPERATING IN A CLUTTERED ENVIRONMENT

5. FUNDING NUMBERS

6. AUTHOR(S) Wee Kiong Ang

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB number ____N/A____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release. Distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

In an increasingly complex environment crowded with obstacles, particularly manned and unmanned traffic, technological advancements can autonomously provide alerts to the presence of incoming threats. In other words, advancements such as computer vision (CV) capability enhance overall situation awareness. This thesis explores the development and integration of CV capability onboard a functional unmanned aerial vehicle (UAV) to detect and track multiple proximate moving targets autonomously.

A systems engineering approach is applied to define, analyze, and synthesize systematically a proposed system architecture for the real-time autonomous detection and tracking capability via visual sensors onboard the UAV. Both the hardware and software architecture design are discussed at length. Then, a series of tests that were conducted progressively to assess and evaluate the overall system architecture are described. Multiple UAVs and unmanned ground vehicles represented the contested operational environment. The developed CV algorithm proved successful at detecting and tracking multiple moving targets in real-time operation, thus laying the foundation for future research and implementation of the developed techniques in the automatic vision-based collision-avoidance guidance architecture.

14. SUBJECT TERMS	15. NUMBER OF PAGES
unmanned aerial system, electro-optics sensor, computer vision, optical flow, situation awareness	137
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540–01-280-5500 Standard Form 298 (Rev. 2–89)
Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

ASSESSMENT OF AN ONBOARD EO SENSOR TO ENABLE DETECT-AND-SENSE CAPABILITY FOR UAVs OPERATING IN A CLUTTERED ENVIRONMENT

Wee Kiong Ang
Captain, Singapore Army
B.S., University of Queensland, 2012

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2017

Approved by: Oleg Yakimenko
Thesis Advisor

Dong Hye Ye
Co-Advisor, Purdue University

Fotis Papoulias
Second Reader

Ronald Giachetti
Chair, Department of Systems Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

In an increasingly complex environment crowded with obstacles, particularly manned and unmanned traffic, technological advancements can autonomously provide alerts to the presence of incoming threats. In other words, advancements such as computer vision (CV) capability enhance overall situation awareness. This thesis explores the development and integration of CV capability onboard a functional unmanned aerial vehicle (UAV) to detect and track multiple proximate moving targets autonomously.

A systems engineering approach is applied to define, analyze, and synthesize systematically a proposed system architecture for the real-time autonomous detection and tracking capability via visual sensors onboard the UAV. Both the hardware and software architecture design are discussed at length. Then, a series of tests that were conducted progressively to assess and evaluate the overall system architecture are described. Multiple UAVs and unmanned ground vehicles represented the contested operational environment. The developed CV algorithm proved successful at detecting and tracking multiple moving targets in real-time operation, thus laying the foundation for future research and implementation of the developed techniques in the automatic vision-based collision-avoidance guidance architecture.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

AO	area of operation
BASH	Bourne Again Shell
BEC	battery eliminator circuit
BIT	built-in-test
CCD	charged-coupled device
CMOS	complementary metal oxide semi-conductor
CONOPS	concept of operations
CV	computer vision
dB	decibel
DHCP	dynamic host configuration protocol
DJI	Da Jing Innovations
DOD	Department of Defense
DODAF	Department of Defense Architecture Framework
EMMI	energy, matter, material wealth and information
EO	electro-optical
FAA	Federal Aviation Administration
FHSS	frequency-hopping spread spectrum
GCS	ground control station
GPS	global positioning system
HD	high-definition
IMU	inertial measurement unit
INS	inertial navigation system
IP	Internet Protocol
ISM	industrial, scientific, and medical
LAN	local area network
LIDAR	light detection and ranging
LiPo	lithium polymer
LOS	line-of-sight
OES	onboard embedded system
OPSIT	operational situation

OS	operating system
P3at	Pioneer 3AT
PDB	power distribution board
POE	projected operating environment
ROS	Robot Operating System
SDK	software development kit
TTL	transistor-transistor logic
UAS	unmanned aerial system
UAV	unmanned aerial vehicle
UGV	unmanned ground vehicle
VATIC	Video Annotation Tool from Irvine California
VO	visual odometry

EXECUTIVE SUMMARY

This thesis demonstrates the feasibility of using visual sensors onboard an unmanned aerial vehicle (UAV) to autonomously detect and track moving targets in real-time operation. This capability enhances situation awareness and reduces the cognitive load of the operators. Unlike previous work using sensors, such as light detection and ranging (LIDAR), the use of an existing lightweight electro-optical camera circumvents the load capacity of the unmanned systems. This thesis work, which is a collaborative effort between Naval Postgraduate School and Purdue University, serves as a foundation for future research and implementations.

Based on the Department of Defense unmanned system roadmap, unmanned systems such as UAVs and unmanned ground vehicles (UGVs) are gaining popularity in all aspects of applications (2005). The growing presence of unmanned systems results in an increasingly complex operating environment, crowded with manned and unmanned traffic (Teo 2013). In addition, this environment places a tremendous cognitive load on pilots who need to handle multiple interfaces. Therefore, it is imperative to leverage technological advancements to alert pilots autonomously of the presence of other proximate moving targets.

This research aims to develop, integrate, and assess the use of autonomous detection and tracking of multiple moving targets via visual sensors onboard UAVs. A systems engineering approach was adopted to facilitate the design and development of the overall system architecture. This approach defined the issue and the boundaries to highlight the context of the design space. The research then investigated the imposed limitations and elaborated on the design reference mission to illuminate the operational context of the system. Functional analysis enabled the author to derive the functional requirements. Thereafter, an overall system architecture that would fulfill the stakeholders' needs and requirements was proposed. This system underwent progressive testing and evaluation to assess the use of such capability in real-time operations.

The proposed system architecture comprises several subsystems. The computer vision (CV) algorithm responsible for the autonomous detection and tracking capability is executed by an onboard embedded system (OES). The OES is integrated onto a functional UAV for real-time operation to identify the moving targets. These moving targets are simulated by UGVs programmed for autonomous waypoint navigation through a series of feedback control actions. A ground control station (GCS) links all the subsystems via a local area network (LAN), allowing the operator to command and control all the connected devices. The overall system architecture is designed to require minimal operator input.

This thesis notes that trade-offs, such as the selection between transmission range or bandwidth for the communication links, must be carefully considered for the design of the system architecture. The trade-off considerations depend on the stakeholders' needs and mission requirements.

The CV algorithm performs a series of processes to isolate moving targets (Li et al. 2016). It assumes that the moving targets are rigid bodies with densely populated points, and that they have a different motion from the background. First, from the video feed inputs, the background motion is estimated via perspective transformation. The background motion is then subtracted from the sequence of video frames to identify the moving objects. The local motion field of these moving objects can be calculated and pruned based on the motion difference threshold. The pruned objects are then classified as targets based on the angular variance and point density threshold. Subsequently, Kalman filtering is performed on these moving targets to enforce their temporal consistency throughout the detection and tracking process. These moving targets are highlighted with bounding boxes, and this annotated video feedback can be streamed to the GCS in real time.

For this research, tests were conducted throughout the various developmental stages. Experiments with varying CV algorithm parameters enabled fine tuning of the detection and tracking process. The system was tested with video feed resolution at 480p and 600p. Classifier parameters were tuned for small and average targets to explore the effects on the sensitivity of the detection. From the evaluation process, it was observed

that higher resolution video feed provides a wider field of view with better clarity. However, due to the increased encoded data and network limitations, high-resolution video may result in increased latency. Classifier parameters tuned for small targets provide for consistent detection and tracking, but over-tuning of the parameters may result in unwanted noise. The developing CV algorithm, which is still in development, faces some challenging artifacts, such as expanding bounding boxes and reduced effectiveness in strong wind and rolling shutter effects.

References

Jing Li, Dong Hye Ye, Timothy Chung, Mathias Kolsch, Juan Wachs, and Charles Bouman. 2016. "Multi-Target Detection and Tracking from a Single Camera in Unmanned Aerial Vehicles (UAVs)." *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems:* 4992–97. New York: Institute of Electrical Electronics Engineers. doi:10.1109/IROS.2016.7759733.

Teo, Harn Chin. 2013. "Closing the Gap between Research and Field Applications for Multi-UAV Cooperative Missions." Master's thesis, Naval Postgraduate School, Monterey, CA.

U.S. Department of Defense. 2005. *Unmanned Aircraft Systems Roadmap 2005–2030*. Washington, DC: Office of the Secretary of Defense.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

This thesis would not have come to fruition without the intricate collaboration and constant support from the hardworking team at Purdue University. I want to express my sincere appreciation to Professor Dong Hye Ye and Jing Li for their hospitality during my learning trip to Purdue University. Jing Li was especially helpful and patient with me throughout the entire course of my research. I am also thankful to Professor Travis Field and Ignacio Hernandez from University of Missouri at Kansas City. Their expertise and professionalism in the field of unmanned systems inspired me to broaden my breadth of engineering knowledge.

My deepest gratitude goes to Professor Oleg Yakimenko and Professor Fotis Papoulias for their guidance and instruction. Their unwavering support and advice helped me overcome several obstacles throughout my learning journey at Naval Postgraduate School. Professor Oleg was a fatherly figure to me, showering me with constant encouragement. I also extend my heartfelt appreciation to Professor Brian Bingham for his dedicated support in everything robotics. His coaching allowed me to develop my research holistically. I am also thankful for the assistance of Professor Barbara Berlitz, Editor Meg Beresik and Thesis Processor Michele D'Ambrosio who helped to structure and sharpen this report.

I am truly grateful to Singapore Armed Forces for providing me this opportunity to further my studies and enrich my life experiences. I am blessed to work alongside the very talented 2016 cohort of Temasek Defense Systems Institute. Finally, I offer my deepest gratitude to my family and wife, Linda, who stood selflessly by my side throughout my entire course of study.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

As defined by the Federal Aviation Administration (FAA) (2017), an unmanned system refers to a system controlled by an operator away from the system via remote control instead of an onboard pilot. Some examples include unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs). Recent technological advancements have brought about a rapid proliferation in the use of unmanned systems (Teo 2016). These systems are widely used by various market segments, including commercial and industrial enterprises, the media, the military, education institutions, and even hobbyists. The Association for Unmanned Vehicle Systems International has estimated that the integration of UAVs into the national airspace system will create 103,776 jobs by 2025, with more than $82.1 billion generated between 2015 and 2025 (Jenkins and Vasigh 2013). The Department of Transportation has projected that more than 250,000 UAVs will be in operation by 2035, surpassing manned aircraft operations. In fact, the Department of Defense (DOD) expects to add more than 14,000 UAVs to its inventory, representing more than 70% of its entire aerial fleet (U.S. Department of Transportation 2013), as depicted in Figure 1.

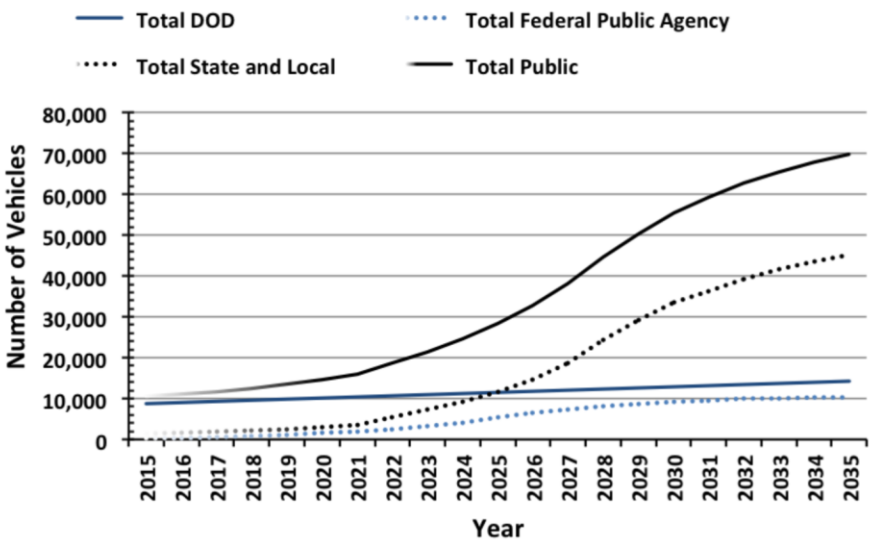


Figure 1. DOD and Public Agency UAS Forecast 2015–2035.
Source: U.S. Department of Transportation (2013).

While technological advancements improve the way of life and spawn new industries, they also cause new challenges to surface. The automobile industry is an example of how technology shaped the transportability of humans, products, and information. However, the increased mobility came at a cost of traffic collisions, fatalities, overcrowding, and pollution (Canis 2015). Such conditions drove the enactment and enforcement of new legislative regulations to ensure public safety. In addition, the development of new sets of technological solutions, such as proximity sensors, can aid in the safe operation of the vehicles (Kon 1998).

Similarly, the rapid proliferation of unmanned systems calls for technical solutions to manage the increasingly crowded space of manned and unmanned systems (Teo 2013). In addition, modern pilots must process an overwhelming amount of information related to the operation of these vehicles and their payloads. This evidently reduces pilots' attention span and situation awareness, leading to higher rates of accidents. To ensure safe operation in such complex environment, the unmanned systems have to perform accurate and timely detection and tracking of multiple objects in their surroundings. Upon sensing the moving objects within their vicinity, the unmanned systems can then safely navigate around these objects.

Previous approaches to the detection of moving objects relied on externally installed sensors such as light detection and ranging (LIDAR) (Opromolla et al. 2016). These externally mounted sensors, however, incur extra load and are not reliable on light sensitive materials. Smaller unmanned systems may not be able to carry the additional weight of these sensors due to their low load capacity. To circumvent the payload restriction, the systems can rely on their existing onboard sensors. All unmanned systems are piloted via some form of visual means; either via a direct line-of-sight (LOS) visual or an onboard electro-optical (EO) camera providing a first-person-view to the ground control station (GCS). These onboard cameras can be used for the detection and tracking of multiple moving objects (Li et al. 2016).

This thesis study paves the way for autonomous detection and tracking of multiple targets while enhancing the UAV payload efficiency. Autonomous highlighting of proximate moving objects can reduce the cognitive load of the pilots. Furthermore, the use of existing onboard camera sensors reduces the demands for additional payload, hence allowing for the deployment of longer endurance and smaller signature unmanned systems.

B. LITERATURE REVIEW

This section details the review of literature from previous studies conducted by scholars and researchers in the field that contributed to this thesis research.

1. Multi-rotor UAV System

UAVs encompass various types of aircraft that a pilot can control remotely. In general, there are three main types of UAVs: multi-rotor, fixed-wing, and helicopter (single or co-axial) UAVs. Each has its own capabilities and serves different functions. A comparison of the different platforms' capabilities is summarized in Table 1. This thesis utilizes the multi-rotor UAV as the research platform.

Table 1. Comparison of UAV Platform Capabilities.

	Scale	Load Capacity	Construct	Endurance	Hovering
Multi-rotor	Small to Medium	Medium	Simple	Short	Yes
Fixed-wing	Medium to Large	High	Simple	Long	No
Helicopter	Small to Large	High	Complex	Long	Yes

The four control inputs for the multi-rotor type are throttle, aileron, elevator, and rudder. Throttles provide the main thrust, ailerons provide the rolling movement, elevators provide the pitching movement, and the rudder controls the yaw heading of the aircraft. These control inputs are used to generate the various kinematics of the aircraft, such as the body frame acceleration (*u*, *v*, *w*) and angular acceleration (*p*, *q*, *r*), which can be mathematically modeled, as depicted in Figure 2.

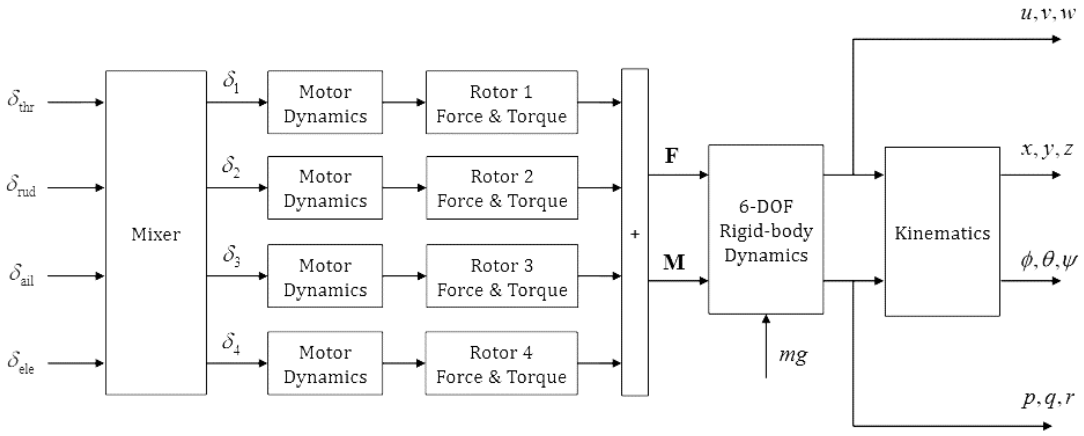


Figure 2. Mathematical Model of Multi-rotor UAV Control.
Source: Phang (2016).

UAV systems usually employ different navigation means such as inertial navigation, dead reckoning, and global positioning system (GPS) navigation. In 2015, Cui et al. published a paper for autonomous navigation of a UAV in a GPS-denied environment. Figure 3 illustrates the navigation system. The UAV performed its state estimation without GPS location using a two-fold process. It first estimated its motion

using LIDAR to generate the velocity estimation. The estimation was then coupled with the inertial measurement unit (IMU) readings, which were subsequently passed through a Kalman filter to derive the position and velocity of the UAV. Graph simultaneous localization and mapping (SLAM) served as a post optimization process to eliminate possible position drift from motion estimate. The estimated state was then used for trajectory planning and flight control (Cui et al. 2015).

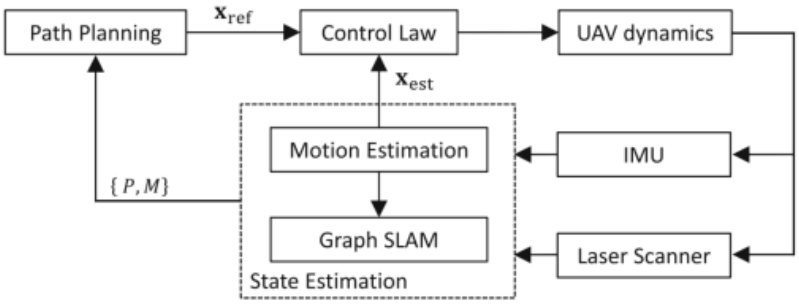


Figure 3. Autonomous UAV Navigation System. Source: Cui et al. (2015).

The team implemented the autonomous navigation system on the customizable UAV platform mounted with two external laser range finders: one to scan the horizontal plane and the other to scan the vertical plane. The UAV platform was equipped with two onboard processors. A single powerful processor was dedicated for computationally intensive mission planning while the other lightweight processor was dedicated for flight control mechanisms. Figure 4 illustrates the schematic layout of the hardware and software of the system.

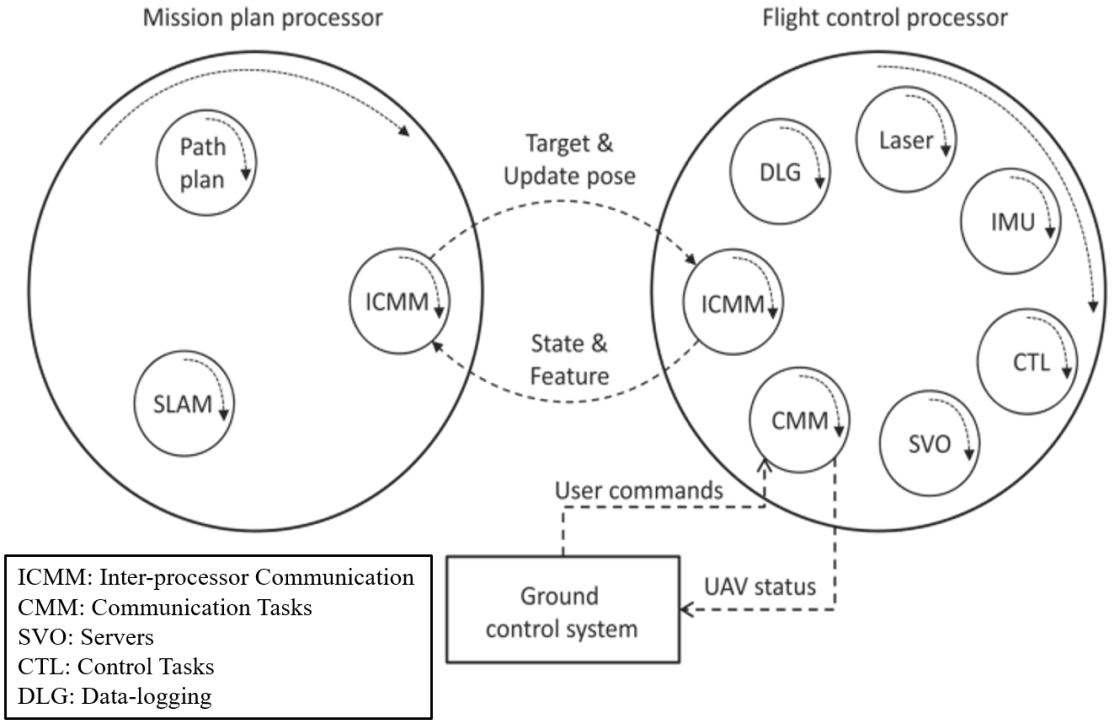


Figure 4. Autonomous UAV Structure Layout. Adapted from Cui et al. (2015).

Instead of using an externally mounted sensor such as LIDAR, this thesis research utilizes an onboard EO camera to generate sensory outputs for the autonomous detection and tracking of multiple moving objects thereby reducing the overall load and allowing the function to be implemented on vehicles with low load capacity.

2. Visual Odometry

To achieve accurate localization, mobile robots must be able to maintain knowledge of their positions. There are several different localization techniques, such as an inertial navigation system (INS), GPS, laser sensors, and visual navigation (Aqel et al. 2016). Visual odometry (VO) is the process of estimating the position of the vehicle by incrementally examining the changes of motion induced by the image capture by onboard cameras. VO is characterized by a good balance between costs and reliability. It is more accurate than INS and other sensors, with relative position error ranging from 0.1 to 2% (Scaramuzza and Fraundorfer 2011). In addition, VO can be used in GPS-denied environment such as indoors or heavily forested areas.

The main challenges in using VO systems are related to the computational power, light, and imaging conditions. There must be sufficient illumination in the environment and sufficient texture to extract apparent motion. Smooth and low-texture areas such as snow pose a challenge for VO to identify salient reference points (Gonzalez et al. 2012). There must be a dominance of static scene over moving objects with sufficient scene overlap between consecutive frames for VO to examine the changes in motion incrementally. In general, VO requires brightness consistency, temporal persistence, and spatial coherence to work effectively (Ang 2016).

Recent technological improvements in processor performance, computational power, and reduced form factor have enabled the implementation of computationally intensive image processing algorithms in real-time operation. This has sparked an increased interest in the field of vision-aided navigation onboard UAVs, which has numerous beneficial applications in disaster control, surveillance, and reconnaissance (Lee and Kwak 2014).

Optical flow is a VO technique used to extract the 2D motion of image objects between two consecutive frames, with the assumption that the pixel intensities of an object do not change between consecutive frames and that neighboring pixels have similar motion (Horn and Schunck 1981). As mentioned by Horn and Schunck, the optical flow equation that represents the change in image object over time is described with I_x, I_y and I_t representing the image gradients, u and v representing the motion vectors, and δt representing the change in time.

$$I(x, y, t) = I(x + u\delta t, y + v\delta t, t + \delta t) \tag{1}$$

Horn and Schunck (1981) further simplified this equation by taking the Taylor series approximation of the right-hand side of the equation:

$$I_x u + I_y v + I_t = 0 \tag{2}$$

Two unknown variables (u, v) cannot be solved with one equation. Collins (2007) describe this constraint as the common Barber Pole illusion where the direction of flow parallel to an edge is unknown. Since the assumption is that all neighboring pixels will have similar motion, Lucas and Kanade (1981) took the various neighboring points

around the point of interest and thus solved the equation using a linear least square estimate. The neighboring points are represented by q_i in the following equation:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x(q_i)^2 & \sum I_x(q_i) I_y(q_i) \\ \sum I_x(q_i) I_y(q_i) & \sum I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum I_x(q_i) I_t(q_i) \\ \sum I_y(q_i) I_t(q_i) \end{bmatrix} \quad (3)$$

The accuracy of the Lucas-Kanade optical flow method can be improved by using the coarse-to-fine iterative refinement (Cohn et al. 2012). First, the Lucas-Kanade equation can be used to estimate motion of each pixel. The motion flow then serves to warp the Image *I* to Image (*I* + *t*) down the image pyramid. This process repeats until it converges to a local minimum. If the underlying transform is significant, then coarse-to-fine refinement will emphasize the large motion as illustrated in Figure 5. If the underlying motion is small, the refinement will initialize with zero (Liu 2011).

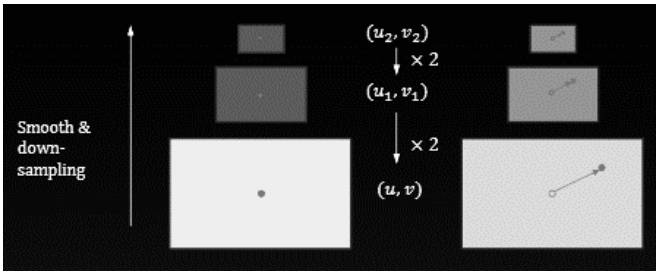


Figure 5. Coarse-to-Fine Refinement. Source: Liu (2011).

In 1994, Shi and Tomasi expanded on the Lucas-Kanade optical flow method by providing a tracking algorithm based on a model of affine image changes. It monitors the quality of image features by using a measure of feature dissimilarity between the first and current frames. When the dissimilarity between the first and current frame grows too large, the algorithm abandons the feature (Shi and Tomasi 1994).

This thesis research utilizes the Lucas-Kanade Optical Flow and Shi-Tomasi Corner Detection method to detect and determine the optical flow of the moving objects, before implementing the Kalman filtering for tracking.

3. Multiple Robot Control

Besides focusing on the development and integration of the computer vision algorithm onboard a UAV, this thesis demonstrates the control of multiple UGVs as the proxy for the moving targets that will be detected and tracked. As described by Parker (1993), the common types of robot interactions to achieve distributed intelligence for multiple robot control are coordinative, collective, cooperative, and collaborative.

In coordinative systems, the robots are aware of each other, but they do not share a common goal, and their individual actions are not beneficial to the others. By contrast, robots in collective systems are unaware of each other's presence, but they share the same goal and have helpful actions to each other. In cooperative systems, robots are aware of each other, share the same goal, and their actions are beneficial to their teammates. Robots are also aware of each other in collaborative systems, but they have individual goals while their actions are beneficial to the others. The various types of interactions are illustrated in Figure 6.

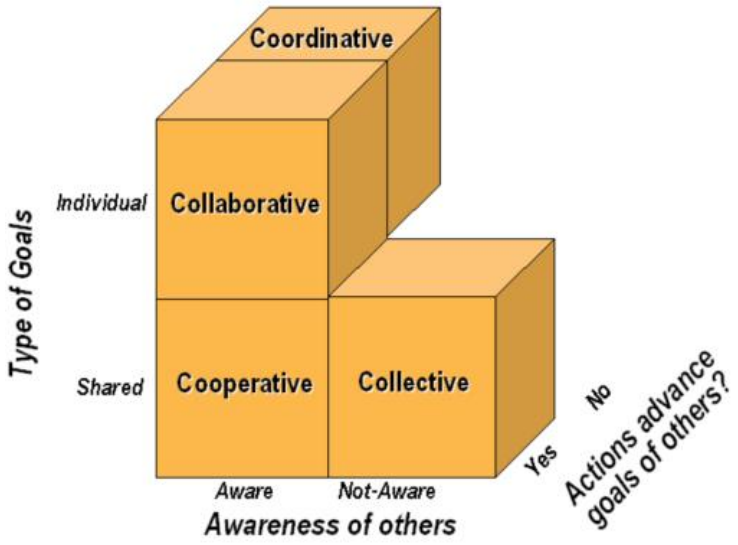


Figure 6. Categorization of Robot Interactions. Source: Parker (1993).

The teams of multiple UGVs used for this thesis research are entities of a cooperative control system. They were governed by both local and global control laws.

Local control laws guide the action of the individual UGV based on the information derived from its own sensors, thereby allowing the UGV to react to its environment without pre-conceived notions of the surroundings. Global control laws utilize the team goals, which cannot be sensed in the UGV's own local domain, to direct the individual UGV actions, (Parker 1993). By balancing the use of local and global information, the teams of UGVs were able to travel in a pre-determined formation without excessive communication requirements.

In 2002, Fierro et al. expounded on the multi-robot cooperative control problem by identifying two main areas of research: formation control and trajectory generation. Formation control consists of three main types: leader-following, behavioral, and virtual structure. In a leader-following setup, one robot (leader) generates the reference trajectory for the rest (followers). Behavioral setup prescribes basic behaviors such as obstacle avoidance and goal seeking for the robots. The virtual structure approach considers and derives the control dynamics of the entire group of robots as a rigid structure. Thus, formation control provides the robot with the appropriate behavior.

Trajectory generation can be achieved by the use of a potential field method to construct repulsive field forces around obstacles and attractive field forces around the goal location, thereby generating a reference trajectory for the robot. Fierro et al. (2002) summarized the formation control and trajectory generation into a framework, as illustrated in Figure 7.

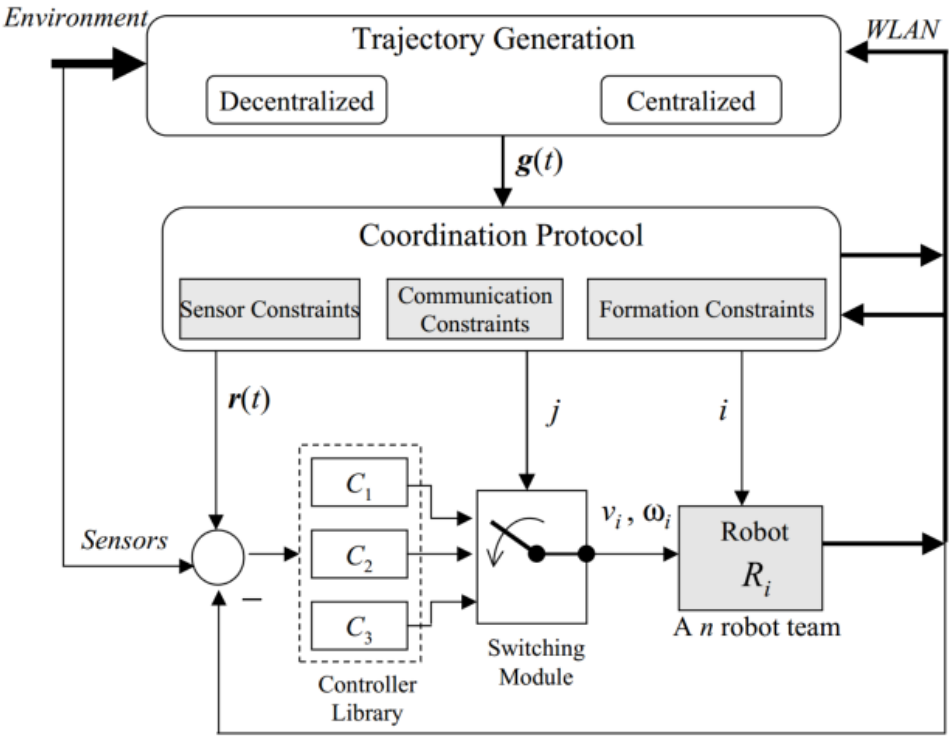


Figure 7. Multi-Robot Cooperative Control Framework.
Source: Fierro et al. (2002).

4. Challenges of Deploying Unmanned Systems

Although unmanned systems are readily deployed for a wide variety of applications, engineers and operators encounter numerous challenges in the design, integration, and operation of these systems. This section discusses some of the potential issues raised by the literature sources.

a. Power Supply

The general power supply for unmanned systems is either via batteries or via combustion engines. While the former generally provide less power output than the latter, batteries are preferred for their modularity. Moreover, they pose fewer logistical and safety concerns. Lithium polymer (LiPo) batteries are widely used in unmanned systems as they are rechargeable and have a higher energy density compared to nickel metal hydride batteries (Han and Lim 2013). The power consumption drives the number and the capacity of batteries used. The engineers need to seek a balance between the overall

carrying load capacity and power consumption for operation and payload requirements. It is not surprising, then, that battery and energy storage have been heavily researched topics in the recent years.

b. Communication Means

All unmanned systems require wireless communication with the GCS and operators. For effective remote operation, the latency of the communication needs to be low. Wireless signal transmission, however, requires security encryption to prevent possible interference and deliberate attacks, such as spoofing, which may cause delays. Some unmanned systems are operated over long distances, thus requiring low frequency signals or over-the-horizon communication. With the increasing resolution ranging from 1080p full high-definition (HD) to 4K quality, unmanned systems are requiring a higher bandwidth for data transmission.

WiFi most commonly provides widespread access networks for connecting the end users' wireless devices. In recent years, the WiFi standard family, as specified by IEEE 802.11 a/b/g/n/ac, has increased its transmission rate significantly using the 2.4 GHz and 5 GHz industrial, scientific, and medical (ISM) radio bands. WiFi provides two different modes of operation, ad-hoc mode and the infrastructure mode. In 2016, Brown et al. highlighted that ad-hoc mode consumes less power and is more responsive for peer-to-peer connection among a few devices while infrastructure mode provides better performance in terms of signal strength and bandwidth for supporting connections among numerous devices. Nevertheless, ISM is already crowded with numerous applications and appliances. This reduces the signal-to-noise ratio and the link quality. Therefore, engineers have to consider the tradeoffs related to distance of transmission, bandwidth, and security.

C. PROBLEM FORMULATION

To ensure the safe operation of unmanned systems in modern complex environment, this thesis strives to answer two critical research questions:

1. Can the UAV perform accurate and timely autonomous detection and tracking during its flight?

2. Will the computer vision algorithm work in a complex operating environment with multiple moving objects?

This thesis examines the integration of the CV algorithm onboard a UAV to autonomously detect and track multiple moving targets. To guide the reader, the thesis is organized correspondingly with the various development stages of the research.

Chapter II discusses the systems engineering approach adopted by the author for the development of the UAV and the CV system. This section defines the research problem and identifies the boundaries of the system. It describes the functional and physical decompositions in adherence to the Department of Defense Architecture Framework (DODAF). This chapter also presents the proposed system architecture by describing the various unmanned systems and subsystems used, with justifications for the selection of the various system components.

Chapter III discusses the technical design of the overall system by examining the implementation of the CV algorithm on an Onboard Embedded System (OES), its mechanism, and the key parameters governing the behavior of the algorithm. This chapter also explains the process of achieving autonomous waypoint navigation in the UGVs.

Chapter IV presents the conduct of progressive experiments to test and evaluate the overall system architecture. This chapter also discusses the results of the experiments and the collected data. Chapter V summarizes the thesis with conclusions and recommendations for future work.

THIS PAGE INTENTIONALLY LEFT BLANK

II. SYSTEMS ENGINEERING APPROACH

Buede, Dennis, and Miller describe a system as "a collection of hardware, software, people, facilities, and procedures organized to accomplish some common objectives" (2016, 3). As defined in Annex G of ISO/IEC/IEEE 15288, a system of systems comprises of interacting systems to perform a function that is greater than the sum of the individual systems. The unmanned system itself is a complex system comprising numerous other sub-systems and components; this system is a purposeful whole consisting of interacting parts. In addition, the unmanned system interacts in an evolving ecosystem consisting of other external systems and environments. The systems engineering approach provides a formal methodology to study comprehensively the interactive nature of a system of systems.

DOD's *Defense Acquisition Guidebook* (2000) defines systems engineering as an integrated effort across multiple technical disciplines to bring out balanced solutions that fulfill customer needs. Similarly, the International Council on Systems Engineering (INCOSE) summarizes systems engineering as a comprehensive approach to realize successful systems that will meet the requirements of stakeholders. The systems engineering approach is used to explore the potential characteristics that influence the development of the unmanned system used in this research.

This chapter also presents the system architecture of the unmanned system used for the thesis research. In 2012, Hilliard defined system architecture as the fundamental concept that details the characteristics and specifications of a system, as well as its inter-related elements and their relationships. He used DODAF to illustrate the various viewpoints of the system infrastructure.

A. SCOPE OF THE PROBLEM

To focus on the effort needed to achieve the intended purpose and solution of this research, it is imperative first to define the problem. Problem definition includes the analysis of needs, the stakeholders, and the boundaries of the system.

1. Needs Analysis

Needs analysis translates the primitive needs identified by the stakeholders into an effective need statement that answers the pertinent problem. Primitive needs are unsupported opinions while effective needs are derived from criteria and supporting evidence. Currently, no solution exists for real-time autonomous detection and tracking of multiple moving objects.

The Ishikawa diagram, also known as a cause-and-effect diagram, is often used to investigate and identify the root causes of a problem (Wong 2011). In the present research, the Ishikawa process was used to analyze the problem by decomposing it into the aspects of people, policy, information, and resources. Refer to Figure 8.

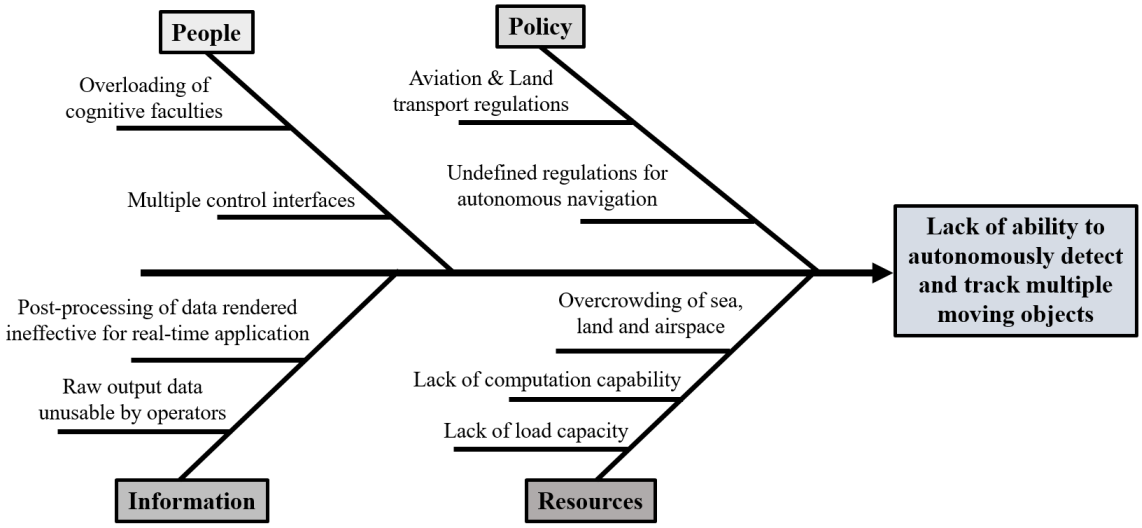


Figure 8. Ishikawa Cause-and-Effect Definition of the Problem.

In Figure 8, "people" refers to the operators of the unmanned systems as those who are burdened by having to use multiple interfaces. Furthermore, the operators' cognitive faculties are overloaded with information and the need to process multiple inputs. Policies, such as the FAA regulations, confine the operation of unmanned systems to a certain area of operations (AO). There are also currently no mandates on autonomous

navigation to guide the development of autonomous detection and navigation of the unmanned systems.

As shown in Figure 8, information from the detection outputs generated onboard the unmanned systems are post-processed from recorded video footages. These detection algorithms serve as technological stepping stones, but they lack real-time application in the operation of the unmanned systems. Additionally, some of these outputs are in raw data format, which cannot be used directly by the operators. While some unmanned systems do not possess the resources to process computationally intensive algorithms, others might not have the load carrying capacity to integrate additional onboard processors.

With the overcrowding of the sea, land, and airspace, there is an effective need to develop and integrate real-time autonomous detection and tracking capability onto the UAV. Successfully addressing this effective need is the goals of this thesis research. Specifically, the UAV must be able to track multiple moving objects, simulated in this research by the UGVs navigating autonomously via their waypoints. This is an effort to demonstrate safe autonomous navigation of multiple unmanned systems within a complex AO.

Stakeholder analysis was conducted to understand the expectations of stakeholders and to help incorporate their requirements into the development of the system. This analysis involves mapping the stakeholders on a power/interest matrix (Mathur et al. 2007). Power relates to the stakeholders' influence over the project or its outcome, while interest refers to the stakeholders' concern over the progress of the project. In addition, the relationships between the stakeholders are represented by the presence of lines on the matrix, with the heaviest line representing the amount of influence that particular stakeholders have over others. The power/interest matrix is illustrated in Figure 9.

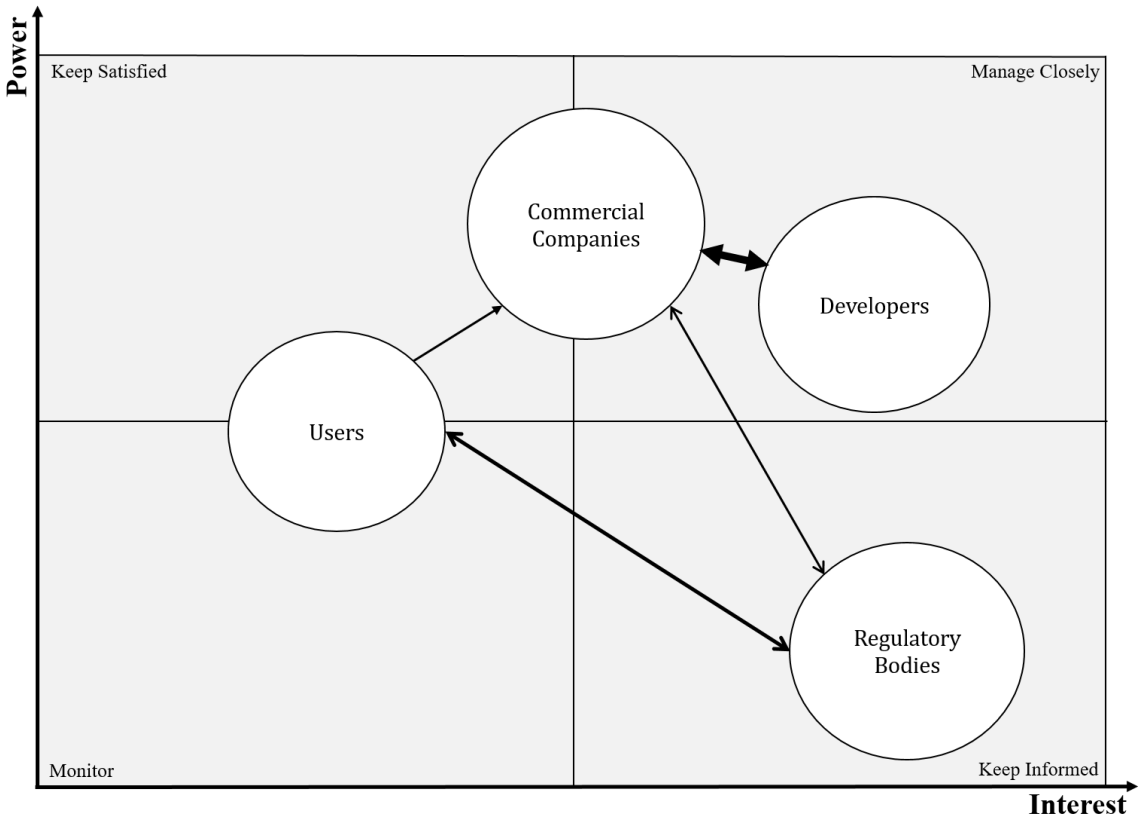


Figure 9. Power/Interest Matrix for Stakeholders' Analysis.

The stakeholders can be categorized based on their influence and interest. They are 1) users, 2) developers, 3) commercial companies, and 4) regulatory bodies. Each of these stakeholder categories is described in the following paragraphs. Table 2 summarizes the goals, concerns, and influences of the key stakeholders.

a. Users

This category includes individuals and organizations with varied requirements that will use the real-time autonomous detection and tracking capability. Users include military and security forces, commercial and media enterprises, education and research institutions, and general hobbyists. Examples of usage extend from surveillance, reconnaissance, aerial photography to drone racing. Users are concerned with the general suitability of the system capability, such as the reliability of detection and tracking, as

well as the maintenance of the system and availability of system spares. In general, users have a vested interest in and influence over the development of the capability.

b. Developers

The developers are the individuals or organizations with strong interest in and influence over the development of the system. Developers include educational institutions such as universities, research and development agencies such as the Defense Innovation Unit Experimental. They are concerned with the advancement of technologies to reach certain system maturity and readiness level.

c. Commercial Companies

Commercial companies generally anticipate market trends and capitalize on the latest technological advancements. They have strong interest in the development of the system, to dominate the market share with trending products. Commercial companies also have a strong influence on the system's development as they invest heavily in research and development to stay ahead of their competitors.

d. Regulatory Bodies

Regulatory bodies serve to ensure the safe operation and management of the unmanned systems. For example, the FAA oversees the management of airspace, as well as the implementation and enforcement of flight regulations. Regulatory bodies adapt their policies to accommodate technological advancements. They have a strong interest in but limited influence over the development of the system.

Table 2. Summary of Stakeholder Analysis.

Stakeholder	Goals	Concerns	Influence
Users	To ease the operation of unmanned systems in a complex environment.	To have high reliability, maintainability, and availability rates.	Consumer demands shape the technological advancements to fulfill the need.
Developers	To fund research, development, and breakthroughs in the technology frontier. To be established as subject matter experts in a particular field.	To be unable to accomplish the research objectives in a specified time span. To lose credibility.	Pushing the envelope for the development of system.
Commercial Companies	To dominate the market. To generate profitable returns.	To face stiff market competition in similar technologies and products. To strive for a balance between budgeting and avoiding obsolescence.	Investing heavily in R&D attracts talent and promotes general maturation of technology and system.
Regulatory Bodies	To ensure safety of the public from the unmanned systems. To regulate the management of the unmanned systems.	To have outdated policies unable to accommodate the latest technologies.	Educating and enforcing the regulations for the use of unmanned systems. Feedback shapes the development of products from commercial companies.

2. Boundaries and Limitations

The definition of system boundaries and any interactions with external systems precedes the determination of the system's own requirements. Boundaries separate the transition between two factors and mediate the flow of energy, matter, material wealth, and information (EMMI) across the interfaces. In this research, the UAV with integrated real-time autonomous detection and tracking capability is viewed as a subsystem that interacts with external natural and man-made systems.

An External Systems Diagram (ESD) can show explicitly the inputs and outputs between the proposed UAV system and external systems, as well as depict the system's

interactions with external systems. The diagram for the proposed UAV system is presented in Figure 10, with the illustration of the EMMI exchanges between the systems.

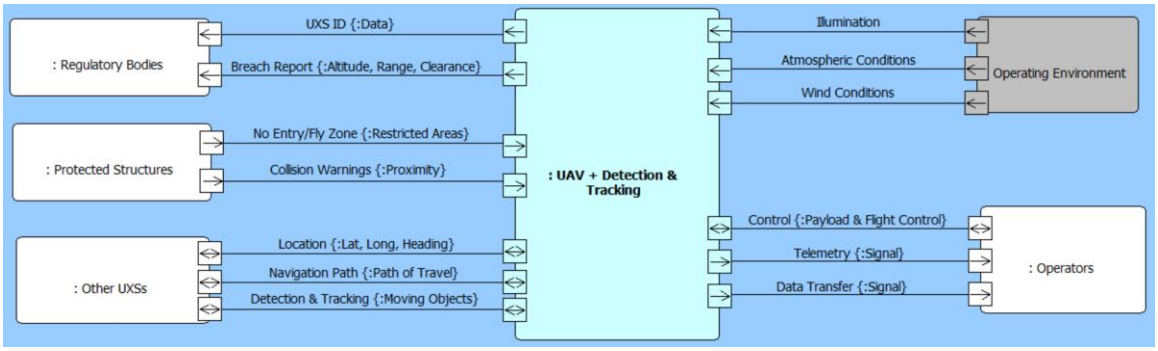


Figure 10. External System Diagram for Proposed UAV System.

a. Natural Systems

Natural systems refer to the various operating environmental conditions. They greatly influence the design and functional baseline of the UAV system. As natural systems are generally difficult to manipulate, man-made systems incorporate technological means to overcome and mitigate these environmental factors to achieve desired performance. The three main natural environmental systems concerning the UAV system are illumination, atmospheric, and wind conditions.

First, the proposed system requires the visualization of moving objects. Thus, illumination in the form of ample lighting is paramount in ensuring the proper functioning of the system. Any variations in luminance level will generate undesired noise, which the UAV system must be able to discern to function flexibly and effectively.

Second, atmospheric conditions affect the operation of the UAV system. Overcast skies with cloud cover may obscure light penetration, thereby affecting the illumination conditions. In addition, a GPS might also be affected if it does not have direct LOS linkage with satellites. In addition, high humidity and particle levels can cause attenuation of signals and distortion of emitted energy. All of these conditions impose consequential effects on overall system performance.

Finally, wind conditions directly impact the operation of the UAV system. High wind conditions may ground the UAV. Furthermore, wind conditions may disrupt the smooth operation of the detection and tracking capability, inducing interference due to the unstable operation of the system.

b. Man-made Systems

In contrast to natural systems, man-made systems are relatively manageable and interactive entities. The various man-made systems that the UAV systems interact with are the operators, regulatory bodies, neighboring structures, and other unmanned systems.

Operators provide the control inputs to the UAV system for the operation of the payload and flight mechanisms. The UAV system sends feedback to update the operators on the system's control status. The UAV system also exchanges telemetry signals to notify the operators of its general status such as battery life and signal link quality. Payload data is also transferred from the UAV system to the operators. This data is used to direct the next set of control inputs.

As mentioned previously, regulatory bodies enforce the regulations on the use of unmanned systems. For example, the FAA is responsible for ensuring and providing a safe aerospace system. The identification number of any unmanned systems that flaunt the rules will be reported to the regulatory bodies, along with a breach report detailing the operating conditions such as altitude, range, and clearance.

Protected structures usually work in tandem with the regulatory bodies to enforce regulations and prevent breaches. Such structures are protected from other systems, including unmanned systems, by an out-of-bounds area. These areas are equipped with collision warning systems should they detect other unauthorized systems in their proximity.

The proposed UAV system also interacts with other unmanned systems. Through the autonomous detection and tracking capability, the UAV system is able to identify other moving unmanned systems and objects. Ideally, the UAV system exchanges its

current location (latitude, longitude, and heading) and its intended path of travel with other unmanned systems.

B. OPERATIONAL CONCEPT

After reviewing the problem, the stakeholders and their needs, and after defining the boundaries, the researcher must establish the Concept of Operations (CONOPS) for real-time autonomous detection and tracking onboard an UAV. The CONOPS is illustrated by the DODAF Operation View (OV-1) in Figure 11. The top-level OV-1 focuses on the behaviors and functions describing the mission of the system. It describes the operational nodes, their tasks, and information exchanges.

Figure 11. OV-1 for Autonomous Detection and Tracking UAV System.

In this framework, the operator provides the necessary control inputs and parameters to the GCS. The control inputs typically consist of waypoints for autonomous navigation or remote control feedback. The operator is able to control multiple unmanned systems using a single GCS. The operator also initiates the real-time autonomous

detection and tracking capability onboard the UAV system. These commands are relayed from the GCS to the UAV system via signal transmissions. Upon receipt of the commands, the UAV system proceeds with its flight operation. The UAV EO camera captures video input throughout the entire flight operation, parsing it to the onboard processor for real-time image processing. The other unmanned systems moving within the field of view of the UAV system are detected and tracked. The information, available in either text or annotated video output, is transmitted from the UAV back to the GCS in near real time. The video feed is also recorded for post-processing purposes. Operators can then access and utilize the information to enhance their mission effectiveness.

C. DESIGN REFERENCE MISSION

Design reference mission (DRM) is utilized extensively by the U.S. Navy to meet 21st century challenges and uncertainties. DRM defines the project operating environment and the specific operationally representative situations baseline for a rigorous systems engineering process (Skolnick and Wilkins 2000).

1. Projected Operational Environment

The projected operational environment (POE) sets the context in which the UAV system will be operated and evaluated for measurable outcomes. This also extends to cover the handling and storage modes. POE considers pertinent aspects of the environmental conditions and regulatory limitations.

a. Environmental Conditions

As previously noted, the UAV system will be operated in naturally illuminated conditions. Thus, the system is subjected to the varying daylight hours set forth by the Winter Solstice, Summer Solstice, Vernal Equinox, and Autumnal Equinox. The POE experiences an annual temperature ranging from 44° F to 68° F. On a monthly average, the POE experiences 30% of cloud cover, 16% precipitation with 1.0 inch of rainfall. Wind speeds of the POE vary from 0 to 25 knots. The expected ground conditions vary from grass field, to concrete floor, to light gravel. The terrain is generally flat and

bounded by low-rise buildings along the perimeter. The unmanned systems are stored at room temperature averaging 60° F with no direct exposure to sun or light.

b. Regulatory Limitations

The FAA imposes peacetime regulations for the flying of UAVs. All UAVs weighing more than 55 pounds must be registered with the FAA. According to regulations, their UAV identification tag must be displayed during flight. Furthermore, the UAV's flight altitude ceiling is capped at below 400 feet, and regulations require pilots to have direct LOS with the aircraft. Recreational pilots are not authorized to fly near airbases, aircraft, or overhead of people. In addition, regulations require all non-recreational pilots to pass aeronautical knowledge testing and possess a remote pilot certification.

2. Operational Situation

Operational situations (OPSIT), generated from CONOPS, are specific instances of a DRM. The design of this research expects two distinct modes of unmanned system operation with varying autonomy: 1) remotely controlled and 2) autonomous navigation. The following paragraphs examine these modes in more detail.

a. Remotely Controlled Operation

The unmanned systems operators have direct control over their systems via the remote control inputs; they manually position their systems to fulfill the mission. Manual control of the systems can increase the cognitive load of the operators, who must be aware of numerous concurrent updates, along with the precise maneuvering of their systems in a crowded operating environment. The real-time autonomous detection and tracking capability serves to enhance the situational awareness of the operators without adding to their cognitive load. The proposed UAV system is able to identify multiple moving objects and highlight them for the operator. The operators can then smoothly execute their mission, whether it is combat engagement or general avoidance. In this OPSIT, the multiple moving objects are multiple remotely controlled UGVs.

b. Autonomous Navigation

The unmanned systems can be set to navigate autonomously in their operating environment. In such cases, unmanned systems will generally adhere to a pre-determined travel path along the waypoints. Autonomous navigation is useful for "dull, dirty and dangerous" missions, such as surveillance and reconnaissance, to reduce the risk exposure of human operators. Upon commencement of autonomous flight, the detection and tracking capability is initiated to provide near real-time feedback to the operators. If moving objects are identified, the operators can override the UAV system to manually redirect the UAV to fulfill its mission. In this OPSIT, the moving objects are multiple UGVs set to navigate autonomously about the pre-determined waypoints.

c. Commonality in OPSITs

Although the level of autonomy differs for the various modes of operation, the OPSITs have common operational activities, as shown in Figure 12. DODAF Operational View 2 (OV-2) graphically depicts the operational nodes and their exchange of information. In this figure, need-lines illustrate the connections between the nodes. The common operational nodes are the proposed UAV system, operators, and moving UGVs. The UAV system controlled by the operators will conduct near real-time autonomous detection and tracking of the moving UGVs. The information will be transmitted back to the operators for usage. As Figure 12 indicates, signal interference may exist between the control of the UAV and UGVs systems.

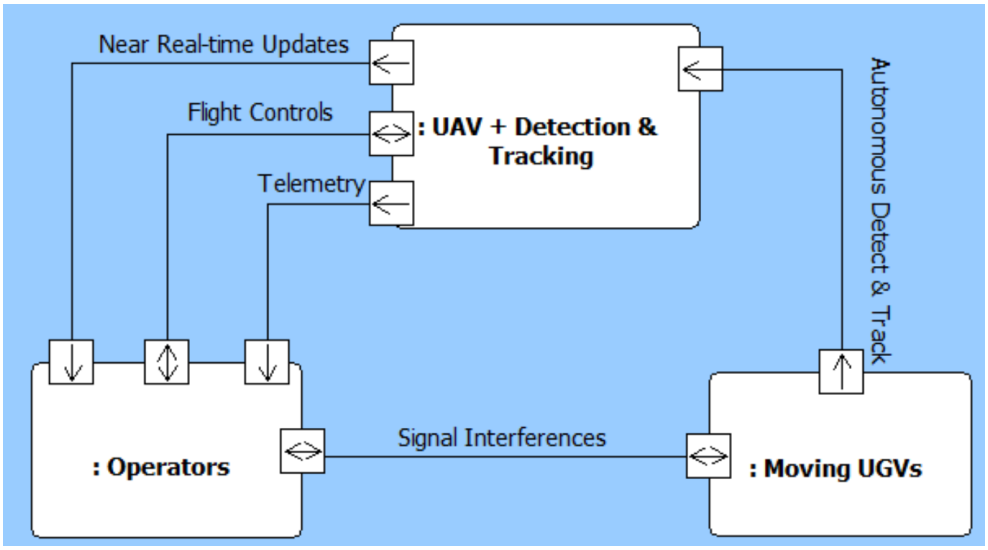


Figure 12. OV-2 for Autonomous Detection and Tracking UAV System.

Upon derivation of the generalized OV-2, DODAF Operational View 5 (OV-5) is generated to delineate the lines of responsibility for the operational activities coupled within OV-2. OV-5 describes the operational activities in detail, with an emphasis on the input and output (I/O) flows between the activities, and is a key product for describing capabilities. OV-5 is illustrated in Figure 13.

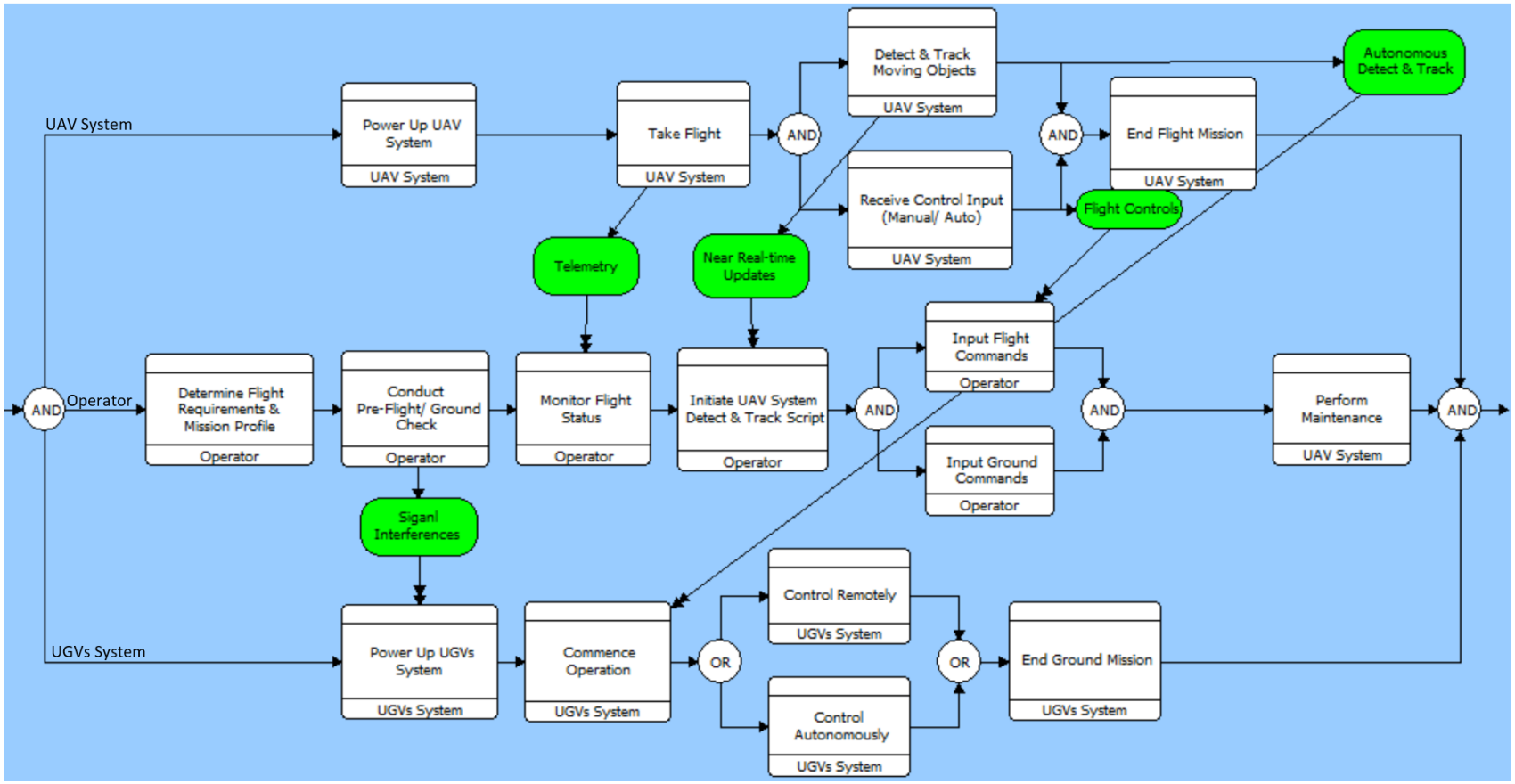


Figure 13. OV-5 for Autonomous Detection and Tracking UAV System

.

D. FUNCTIONAL ANALYSIS

After the determining the UAV system operational activities in OV-2 and OV-5, one can derive its system functions via functional analysis. Blanchard and Fabrycky (2012) describe the conduct of functional analysis as an iterative process that decomposes the complex system requirements into a finer level of abstraction to identify the functions required to operate the system.

System functional decomposition, a key process in functional analysis, can be represented by DODAF System View 4a (SV-4a), which documents the system functional hierarchies at an appropriate level of detail. The decomposed functions are described in the following sections and illustrated in Figure 14.

1. Power up System

Power must be provided to operate the UAV system and the onboard processor responsible for the image processing capability. For increased autonomy, the UAV system must be able to monitor and update its status to the operator.

a. Provide Power to UAV

The UAV system shall be able to provide 22.2 V and a minimum of 4500 mAH 99.9 Wh of power to the system. The battery shall not exceed 600 g to enhance the load capacity of the UAV system, and to protect against electrical short circuit, the system shall provide power cut-off.

b. Provide Power to Onboard Processor

The system shall be able to distribute power efficiently across a range of operational voltages, powering different devices with varying power requirements without any interruption. The UAV system shall also be able to provide 5 V and a minimum of 4000 mAH 20Wh of power to the onboard processor.

c. Conduct Built-In Tests

The system shall be able to conduct built-in tests (BIT) on startup and periodically during operation to ensure proper functioning of the system. In addition, the system shall report the BIT results to the operator for further actions.

d. Transmit Telemetry

The UAV system shall be physically detached from the operator and the GCS. Thus, the system shall be able to transmit its telemetry status, such as battery life and link quality, to the GCS for the operator's awareness.

2. Maneuver in Flight

The UAV and UGVs must traverse within the AO to fulfill their mission profiles. To achieve maneuvering, these unmanned systems shall be able to negotiate obstacles and stay on the projected travel path.

a. Provide Thrust and Propulsion

The UAV system shall be able to provide a mechanical means of power to sustain its movement across the operating environment. The propulsion method shall be able to accommodate the required load capacity of less than 3600 g maximum take-off weight. In addition, the system shall be able to adjust and control the torque generation for varying velocity requirements.

b. Provide Steering Kinematics

The UAV system shall be able to control and adjust its directional heading depending on maneuver requirements. The UAV system shall also be able to achieve six degrees of freedom (DOF), while the UGV system shall be able to achieve two degrees of freedom on an all-wheel drivetrain to negotiate rugged terrain.

3. Conduct Flight Control

The proposed unmanned system will be controlled by the operator either remotely or via autonomous navigation. The inputs from the operators will be transmitted and executed by the system.

a. Determine GPS and Compass Bearing

The system shall be able to determine its location accurately based on GPS means. In addition to knowing its global location, the system must be able to determine its local bearing based on compass heading. Furthermore, the system shall continuously update the GPS and compass bearing at a frequency less than 1 Hz, and update the operator via telemetry means.

b. Accept Remote Control Input

The system shall be able to accept, process, and execute the directional and velocity inputs from the operator via signal transmission. The system shall also continuously update its current location with the inputs from the operator.

c. Execute Autonomous Navigation

The system shall be able to navigate autonomously based on the projected path determined by waypoint inputs, with a deviation threshold of less than 1 m. The function of autonomous navigation is coupled closely with the function of GPS and compass bearing and signal transmission.

4. Detect and Track Autonomously

The onboard processor will be responsible for the autonomous detection and tracking of multiple moving objects. It shall be equipped with sensors to capture the video input for image processing, with tolerance of varying environmental factors such as illumination.

a. Subtract Background Motion

The system shall be able to estimate for the background motion by fitting the local motion field to a global transformation. The background motion is induced by the body movement of the proposed system.

b. Determine and Prune Moving Objects

The system shall be able to extract and identify the moving objects by compensating for the background motion. In addition, the system shall be able to prune the identified moving objects based on a determined threshold to enhance the fidelity of detection.

c. Cluster and Track Targets

The system shall be able to discern the moving objects and prevent intermittent misdetection and false alarms. The system shall also be able to predict and track the path of the moving objects.

5. Communicate Externally

As part of the system of systems, the unmanned system needs to be able to communicate with other operational nodes, such as the GCS, for the transmission of data and control signals.

a. Receive Wireless Signals

The system shall be able to receive wireless signals across the 2.4 Ghz ISM and 5.8 Ghz ISM bands, within the operating ranges of the various frequencies.

b. Process Wireless Signals

The system shall be able to demodulate the analogue signals to digital signals for processing. The system also needs to decrypt the spread-spectrum radio transmissions, such as the direct-sequence spread spectrum (DSSS) or the frequency-hopping spread spectrum (FHSS).

c. Transmit Wireless Signals

The system shall be able to transmit the wireless signals across the 2.4 Ghz ISM and 5.8 Ghz ISM bands, within the operating ranges of the various frequencies. In addition, the system shall be able to transmit based on the required spread-spectrum transmission modes.

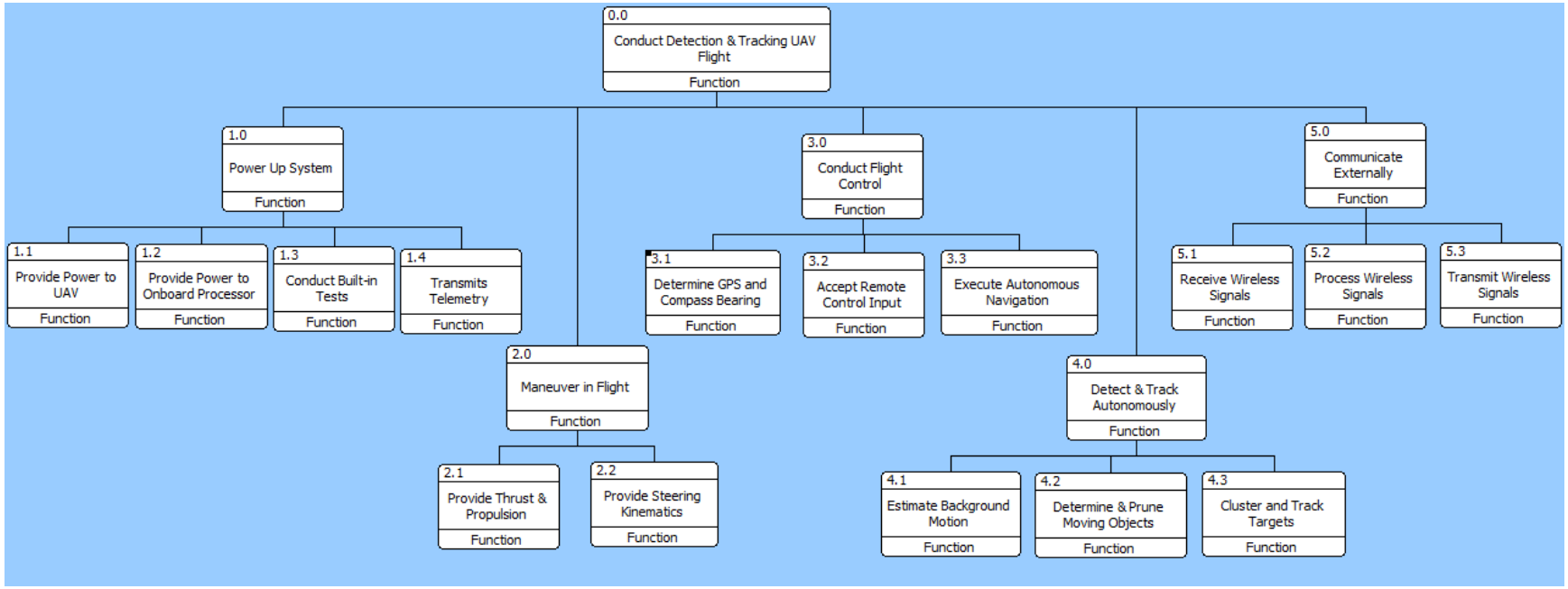


Figure 14. SV-4a for Autonomous Detection and Tracking UAV System.

In addition, DODAF System View 5a (SV-5a), as shown in Table 3, is used to map the operational activities to the system functions to ensure traceability. As described in DODAF version 2, SV-5a identifies the transformation of an operational need into a purposeful action performed by the system.

Table 3. SV-5a for Autonomous Detection and Tracking UAV System.

System Functions			Operational Activities			
			Near Real-time Updates	Flight Controls	Telemetry	Autonomous Detect & Track
			1	2	3	4
1.0	**Power Up System**				x	
	1.1	Provide Power to UAV			x	
	1.2	Provide Power to Onboard Processor			x	
	1.3	Conduct Built-In Tests			x	
	1.4	Transmit Telemetry	x		x	
2.0	**Maneuver in Flight**			x		
	2.1	Provide Thrust & Propulsion		x	x	
	2.2	Provide Steering Kinematics		x	x	
3.0	**Conduct Flight Control**			x		
	3.1	Determine GPS and Compass Bearing		x	x	
	3.2	Accept Remote Control Input		x		
	3.3	Execute Autonomous Navigation		x		
4.0	**Detect & Track Autonomously**		x			x
	4.1	Estimate Background Motion	x			x
	4.2	Determine & Prune Moving Objects	x			x
	4.3	Cluster and Track Targets	x			x
5.0	**Communicate Externally**			x	x	
	5.1	Receive Wireless Signals		x	x	
	5.2	Process Wireless Signals		x	x	
	5.3	Transmit Wireless Signals		x	x	

E. PROPOSED SYSTEM ARCHITECTURE

With the system requirements and functions derived, this research proposes a generic system architecture for autonomous detection and tracking capability onboard a UAV, intended to detect and track multiple moving objects (UGVs). The proposed system architecture is illustrated in Figure 15 using the DODAF System View-1 (SV-1) diagram. SV-1 assigns system nodes to operational nodes, which are described in OV-2. SV-1 also identifies the interfaces between system nodes, translating the need-lines in OV-2 into several interfaces. The proposed architecture consists of four physical sub-systems, denoted by different colors in Figure 15, namely:

1. UAV subsystem - Matrice 100 (depicted in yellow)
2. Onboard Embedded System - Odroid XU4 (purple)
3. UGVs subsystem - Pioneer Robots (orange)
4. Ground Control Station (green)

The following sections outline the hardware and the supporting software utilized by the subsystem, which form the basis of the research thesis.

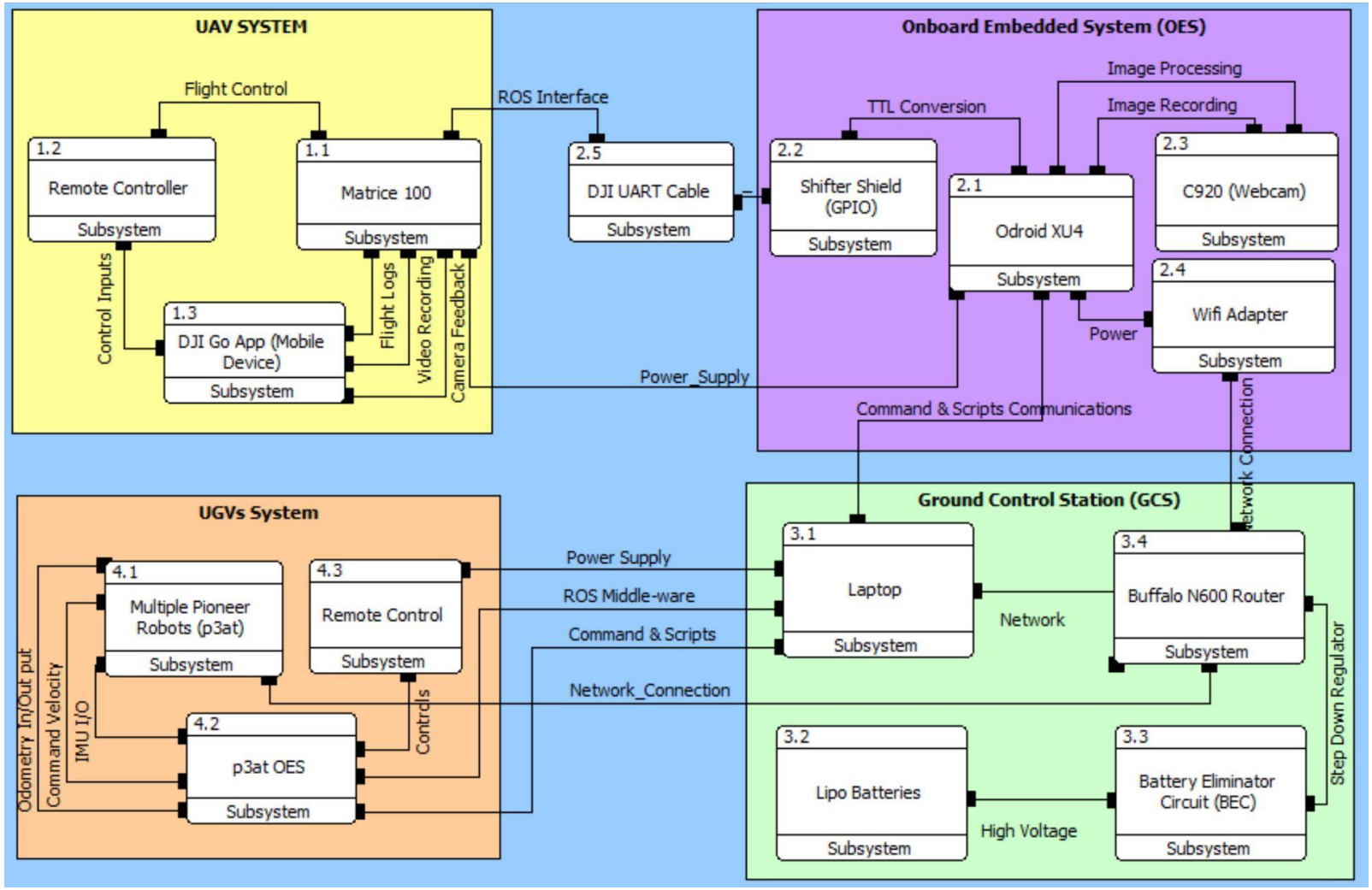


Figure 15. SV-1 of System Physical Architecture.

1. UAV—Matrice 100

The UAV platform used for the thesis research is the Matrice 100 from the Da Jiang Innovations (DJI) Science and Technology Company Ltd. Matrice 100 is an open developmental platform that has a highly customizable architecture, making it suitable for education and research purposes. Figure 16 provides a pictorial overview of the Matrice 100 used in the research.

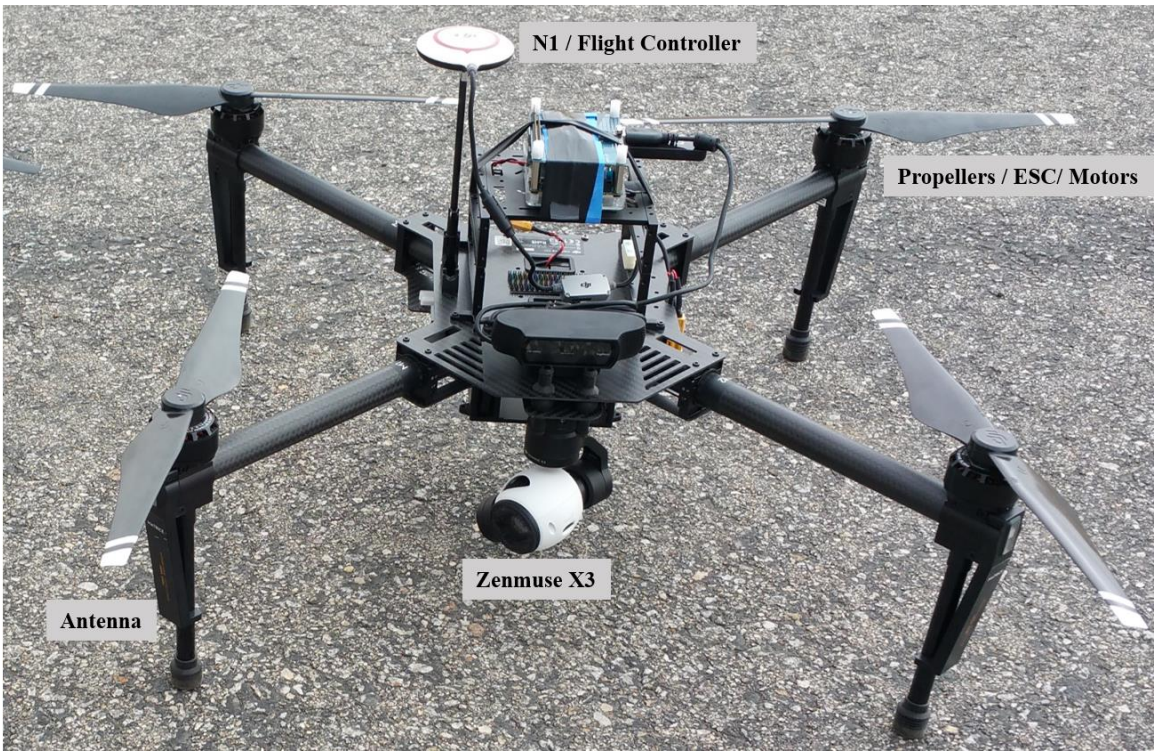


Figure 16. Pictorial Overview of Matrice 100.

a. Hardware Specifications and Configurations

The hardware decomposition of the UAV is depicted in Figure 17. The Matrice 100 is powered by a single TB47D, 6S LiPo battery (22.2 V) with a capacity of 4500 mAh. In total, the power supply sub-system is able to generate 99.9 watts per hour. The power distribution board (PDB) distributes power to support UAV flight as well as other attached accessories and peripherals. The power solutions are depicted in Figure 17a.

The platform is fitted with N1 flight controller avionics that control the movement of the Matrice 100 by integrating the data from the onboard sensors and issuing commands to the propulsion components. The onboard sensors include GPS receiver and INS suite of components, such as compass, gyroscope, accelerometer, and barometer. The flight controller conducts a BIT during system initialization. It provides feedback about the system status to the user interface via wireless communication. The flight controller is also capable of flight planning, generating trajectory, and supporting autonomous flight control based on the input waypoints via the user interface. The N1 Flight Controller and GPS module are shown in Figure 17b.

For the flight-actuator components, the Matrice 100 utilizes four brushless DJI 3510 motors coupled with DJI 1345s propellers and DJI E series 620D Electronic Speed Controllers. These controllers moderate the power supplied from the power distribution unit to the motors, thereby varying the thrust of the motors allowing collective control of the flight motion based on the flight controller. With the generated propulsion, the platform is able to obtain a maximum speed of 22 m/s with no payload and no wind. The platform can generate sufficient thrust to support an extra 1.3 Kg of load capacity. The propulsion system is shown in Figure 17c.

The platform integrates a Zenmuse X3 camera gimbal for target acquisition, as shown in Figure 17d. The onboard high-definition camera sports a Complementary Metal Oxide Semi-conductor (CMOS) sensor with 12.4M pixels, capable of recording 4K quality videos. The three-axis gimbal reads the data from the N1 flight controller, computes the flight movement and inertial forces, and applies the countering angular motion correction to level the camera. It also allows control of the gimbal to point the camera at any 360° view.

The Matrice 100 includes an additional expansion bay for customizing various payloads, which is shown in Figure 17f. Structurally, the platform also has an expandable center frame, and CAN and UART ports to incorporate third-party components and devices, achieving greater functionality.

Communication components of the Matrice 100, shown in Figure 17e, are involved in the transmission of control signals and real-time video feedback. The components include the antennas mounted on the sides of the UAV landing legs and the DJI remote controller C1. The system utilizes both the 5.725 ~ 5.825 Ghz and 2.400 ~ 2.483 Ghz ISM bands. It employs FHSS to coordinate and declutter wireless signal transmissions across numerous devices. The C1 remote controller has an Effective Isotropic Radiated Power (EIRP) of 13 dBm at 5.8 Ghz and 20 dBm at 2.4 Ghz. The key performance specifications of the platform are summarized in Table 4.

Table 4. Key Specifications of the Matrice 100 UAV.
Adapted from DJI Drones (2017).

Parameter	**Specification**
Battery TB47D Voltage / Capacity	22.2V / 4500 mAh
Diagonal Wheelbase	650 mm
System Weight with TB47D	2355 g
Maximum Takeoff Weight	3600 g
Maximum Wind Resistance	10 m/s
Maximum Speed w/o payload and wind	22 m/s
Hovering Time w/o payload and wind	22 mins
Transmission Range (LOS, no interference)	5 Km

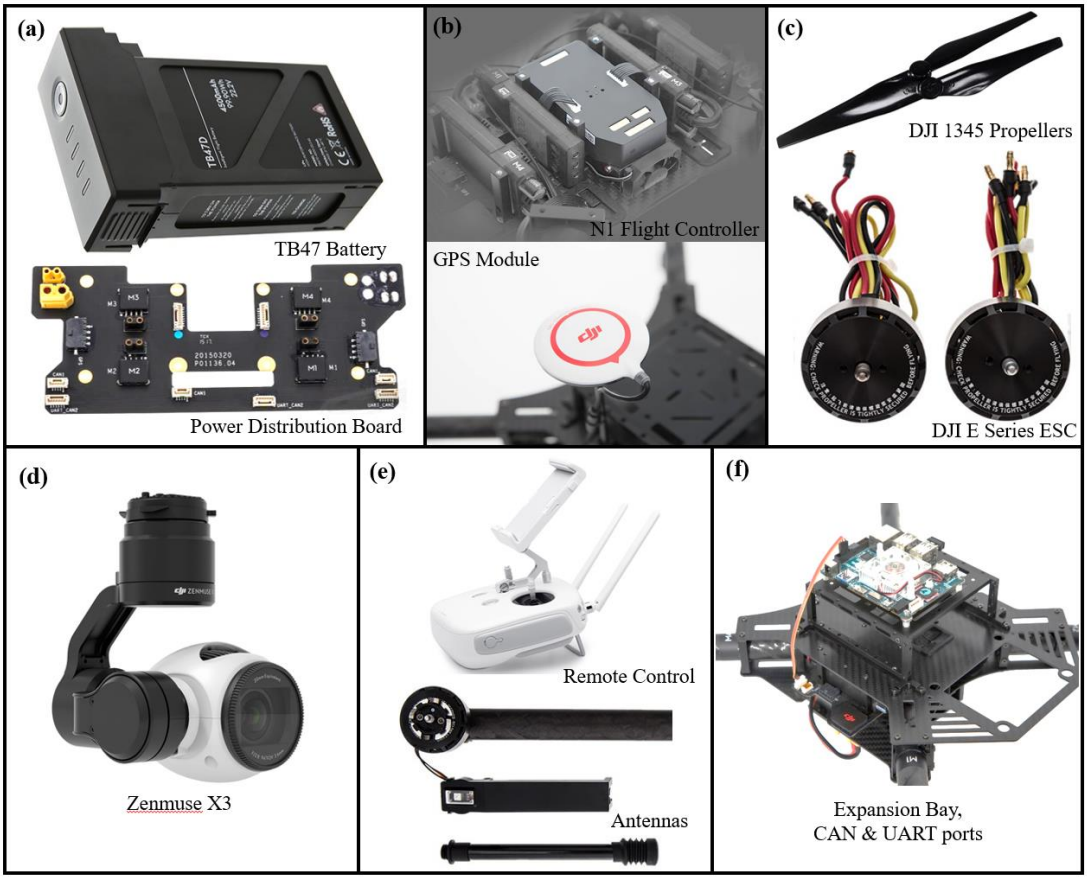


Figure 17. Components of the Matrice 100. Adapted from DJI Drones (2017).

b. Supporting Software

The Matrice 100 is supported by a graphical user interface mobile application, the DJI GO application. This application is connected to the C1 remote controller and linked with the UAV. It allows the operator to interface with the Matrice 100, providing control inputs via remote controller, and receive video feedback from the onboard camera. The research utilized firmware v.1.3.1.00 for the Matrice 100, 1.8.100 for the Zenmuse X3, 1.8.0 + for the DJI Remote Control, and 3.1.6 for the DJI Go App.

The Matrice 100 also supports the application programming interface control feature, allowing customizability via the onboard DJI software development kit (SDK). The DJI SDK utilizes the Linux Robot Operating System (ROS) as a middleware for the implementation of user-created solutions.

2. OES—Odroid-XU4

The platform uses an externally mounted OES, the Odroid-XU4, for image processing functions. The Odroid-XU4 is a powerful single-board computer capable of handling computationally intensive processes. Being a small form factor weighing less than 65 grams, the Odroid-XU4 can be easily incorporated onto the Matrice 100 and other smaller unmanned systems. This OES is fitted with other peripherals, such as a WiFi adapter and a webcam, to perform near real-time autonomous detection and tracking of moving objects to update the operator. The pictorial overview of the Odroid-XU4 is shown in Figure 18.

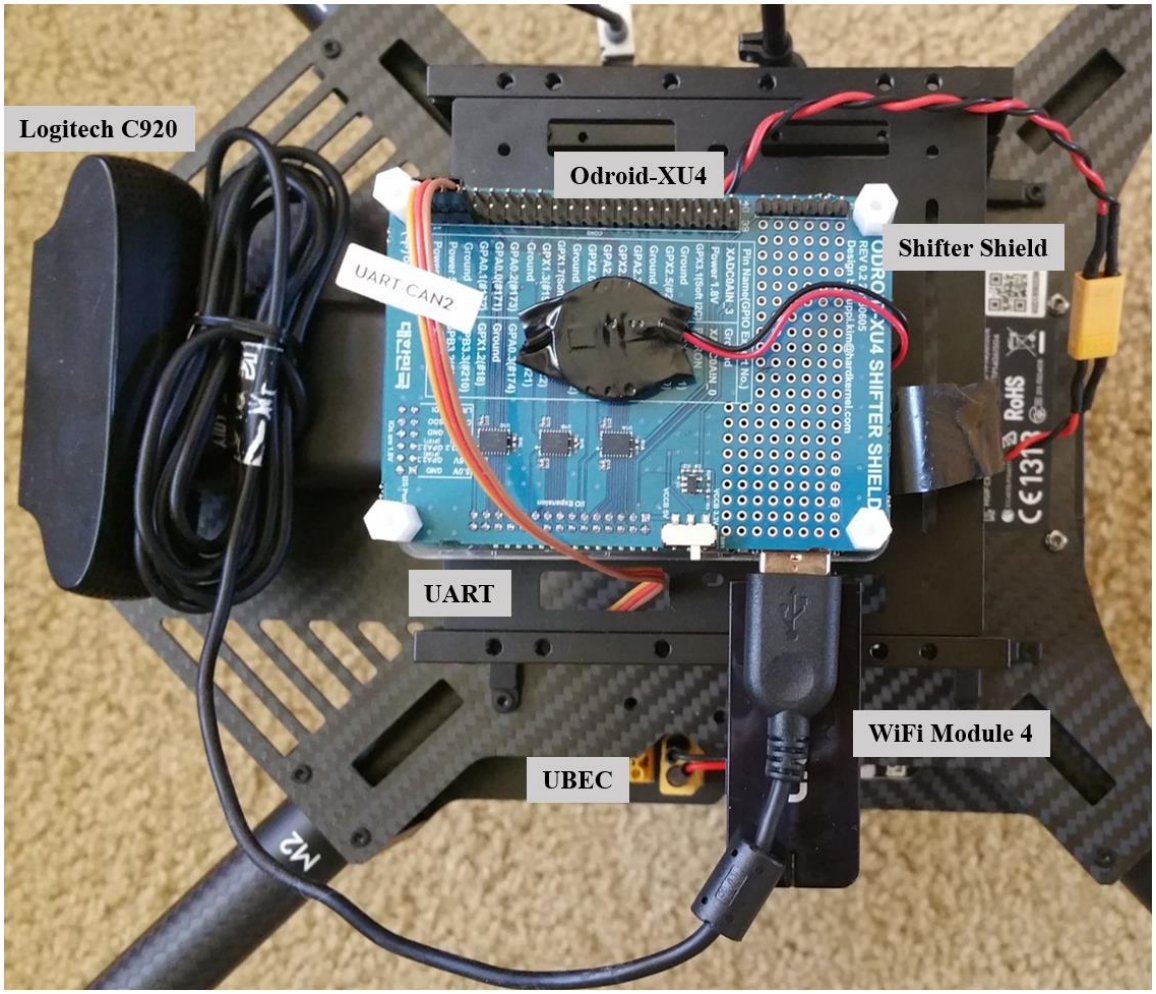


Figure 18. Pictorial Overview of Odroid-XU4.

a. Hardware Specifications and Configurations

The overall hardware decomposition of the OES is shown in Figure 19. The Odroid-XU4 boasts a combination of the Samsung Exynos 5422 Cortex A15 2 Ghz processor and Cortex A7 1.8 Ghz Octa core CPUs. Packed with high-speed two gigabytes LPDDR3 RAM, the Odroid-XU4 has a faster processer relative to other single-board computers. In addition, it implements the eMMC 5.0, USB 3.0, and Gigabit Ethernet interfaces, allowing for high data transfer speeds. Figure 19a depicts the Odroid-XU4 while Table 5 lists its specifications.

Table 5. Key Specifications of the Odroid-XU4.
Adapted from Hardkernel (2017).

Parameter	**Specification**
Dimension (L x W x H)	83 x 58 x 20 mm
System Weight w/o peripherals	< 65 g
Processor Speed	- Cortex A15 2 Ghz - Cortex A7 1.8 Ghz
RAM Speed	Two Gigabyte LPDDR3 RAM
Data Ports	- eMMC5.0 HS400 Flash Storage - 2 x USB 3.0, 1 x USB 2.0 - Gigabit Ethernet port
Required Voltage / Capacity	5 V / 4 A
GPIO Voltage Output	1.8 V

The Odroid-XU4 taps on the power supplied by the Matrice 100 TB47D battery. The PDB from the Matrice 100 distributes 22.2 V to the attached accessories. A Castle Creations 20 A battery eliminator circuit (BEC) Pro, shown in Figure 19b, is installed to regulate the high 22.2 V down to the accepted 5 V for the Odroid-XU4. All the connecting wires used for the Odroid-XU4 setup have an American wire gauge rated for

more than 4 A current carrying capacity. The OES is mounted on top of the Matrice 100 expansion bay to reduce any magnetic disruptions caused by the external devices to the INU and compass sensors.

An EO camera, the Logitech C920 depicted in Figure 19c, is deployed with the Odroid-XU4. Weighing less than 480 g, the webcam is capable of capturing and recording images at a high resolution of full HD 1080p. This webcam utilizes H.264 native video compression, which is compatible with the software scripts. The webcam is also a plug-and-play device with supported drivers installed on the Odroid-XU4, thereby increasing ease of installation. The webcam runs on a CMOS sensor. In comparison with a charged-coupled device (CCD), the CMOS sensor has lower power consumption and is relatively cheaper. However, the CMOS sensor is more susceptible to pixel noise. Furthermore, the CMOS sensor, which utilizes a rolling shutter instead of a CCD global shutter, is prone to rolling shutter artifacts when capturing moving objects at high frequency rates.

The Odroid-XU4 is loaded with onboard DJI SDK with ROS as the middleware. This allows the UAV solutions to be implemented via ROS commands input from the Odroid-XU4. The OES is linked to the Matrice 100 via a UART cable connecting the Odroid-XU4 General Purpose Input Output and the Matrice 100 UART port. All native Odroid-XU4 GPIO operate at 1.8 V but most devices require 3.3 V inputs. A shifter shield with a bi-directional level regulator is installed to level shift the Odroid-XU4 signals to the desired voltage. This resolves the transistor-transistor logic (TTL) de-synchronization, allowing communication between the Odroid-XU4 and the Matrice 100. The UART cable and shifter shield are shown in Figure 19d.

The Odroid WiFi Module 4 adapter, shown in Figure 19e, is attached to the Odroid-XU4, providing networking communication between the OES and GCS. It features a Ralink RT5572N chipset that is natively supported by the Odroid-XU4 drivers. The WiFi Module 4 adapter is capable of IEEE 802.11a/b/g/n WLAN connections using its 2.4 Ghz and 5.8 Ghz dual band antenna. Weighing less than 15 g, the WiFi adapter can establish simultaneous duplex (download and upload) connections at speeds of 57.1 Mbit/s and 105.7 Mbit/s for 2.4 Ghz and 5.8 Ghz, respectively.

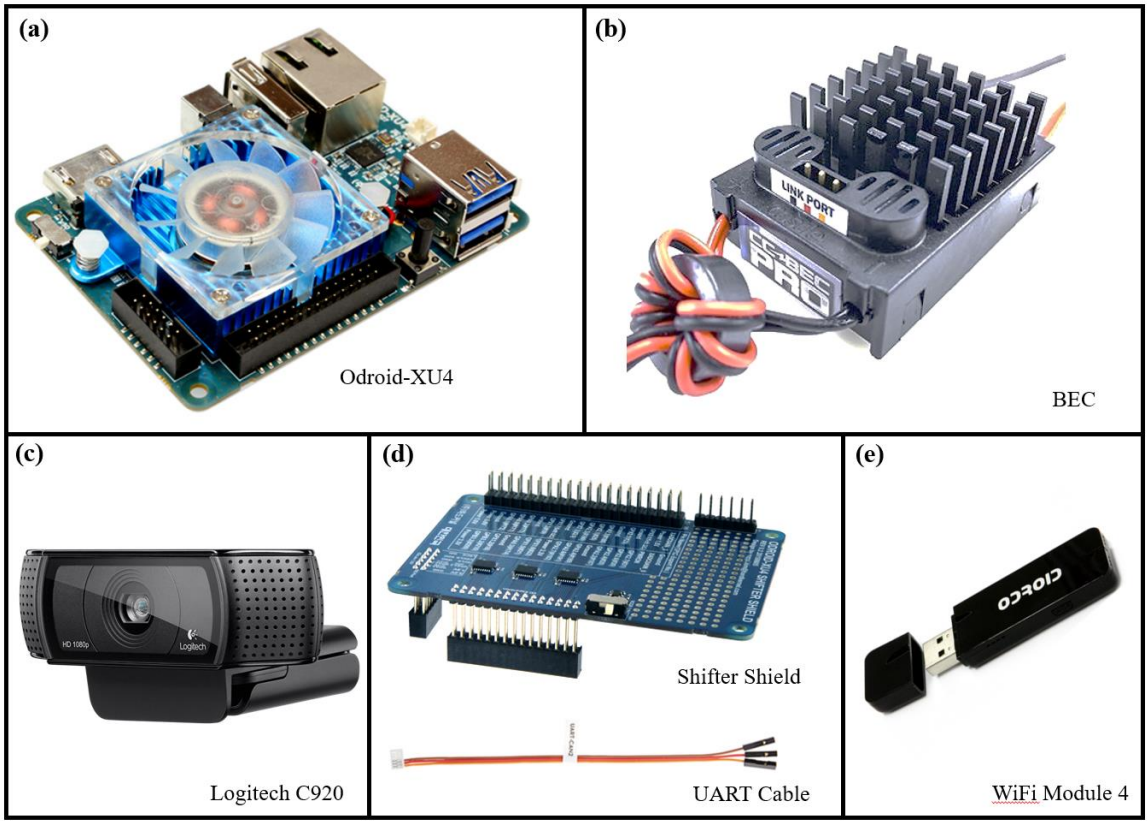


Figure 19. Components of the OES. Adapted from Hardkernel (2017); Logitech (2017); Castle Creations (2017).

b. Supporting Software

The OES is installed with various software to allow the system to leverage the different layers of software capabilities to execute the autonomous detection and tracking capability.

Android 4.4.4 and Ubuntu 14.04 forms the overarching operating system (OS) to support the execution of other software. The OS is a Linux-based open source development that is compatible with most required software and device drivers. Most of the commands are issued via the Bourne Again Shell (BASH) terminal program, which is a keyboard centric means to execute the programs and desired actions.

Python 2.7 provides the main software architecture for executing the autonomous detection and tracking function. Python scripts are used for extracting data from the sensors and performing image processing to identify moving objects. Python 2.7 forms

the core interface for calling on other software, such as Open Source Computer Vision (OpenCV) and ROS, to perform their tasks. Python 2.7 is preferable to other latest releases as it offers stability in cross support with other software. It has comprehensive documentation, community support, and a wider range of compatible script libraries and scientific packages.

OpenCV is installed and compiled as an add-on to Python 2.7. It is designed for computational efficiency and real-time applications of image processing. OpenCV can take advantage of multi-core processing and hardware acceleration to execute computationally intensive programs.

Peripheral programs such as Linux Screen, Wavemon, and Simple Screen Recorder are also installed within the OS environment. Linux Screen allows the use of multiple BASH windows during a single Secure Shell session and it keeps the shells active throughout network disruptions. Wavemon records in real time the link quality of the LAN connection between the OES and the GCS. Simple Screen Recorder is used to record the on-screen display of the GCS laptop for flight analysis and post-processing.

The instructions to flash and set up the Odroid-XU4 are listed sequentially in Appendix A and Appendix B.

3. UGVs—Pioneer 3AT Robots

Pioneer 3AT (P3at) UGVs are used to simulate multiple ground moving targets for the testing of the CV algorithm. The system architecture is designed to require minimal operator inputs, thereby allowing the operators to monitor the CV algorithm feedback. The UGVs can be remotely controlled or perform autonomous navigation within the stipulated AO. The P3at is controlled via its own onboard computer, with commands issued by the operators. A pictorial overview of the P3at is shown in Figure 20.

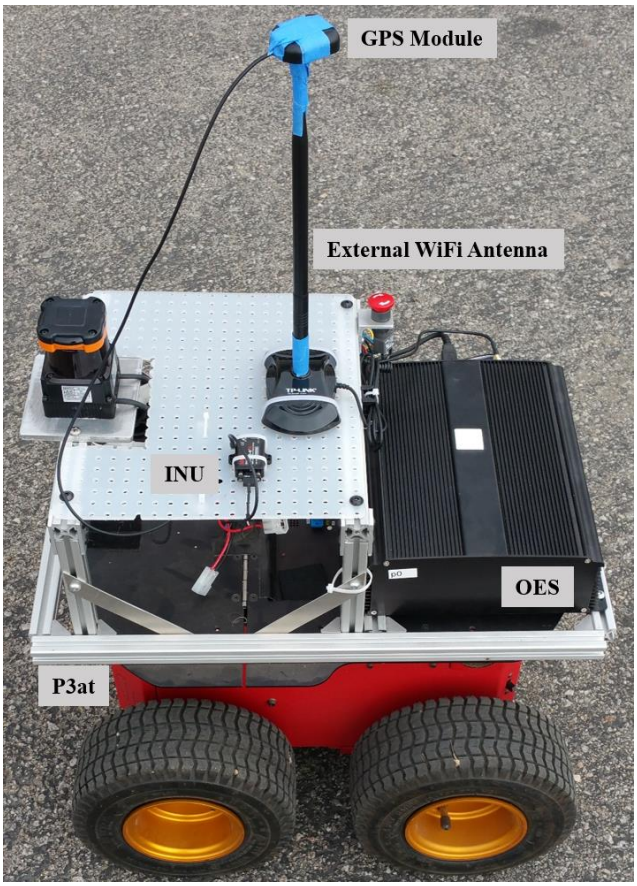


Figure 20. Pictorial Overview of P3at.

a. Hardware Specifications and Configurations

The P3at from Adept Technologies Inc. is a small, lightweight, two-wheel two-motor differential drive robot, ideal for educational and research purposes. Its mainframe is shown in Figure 21a. The P3at is powered by three 7.2 Ah batteries with PDB and voltage regulator to distribute to various power requirements. For navigation purposes, this robot is equipped with wheel encoders to monitor its odometry and heading.

The P3at onboard computer collects and processes the data from the P3at sensors. It is loaded with high-speed RAM and powerful processor capable of generating the desired trajectory and performing the autonomous navigation efficiently. The onboard computer is shown in Figure 21b.

The P3at is equipped with a GPS-aided INU, the Microstrain sensor 3DM-GX5-45, as shown in Figure 21c. The INU provides an all-in-one navigation solution by integrating a range of micro electrical mechanical system INU sensors, such as an accelerometer, magnetometer, and gyroscope with GPS information readings. The INU performs extended Kalman filtering to determine the estimated global position of the robots. In addition, the P3at is fitted with an external TP-Link antenna that provides 8 dBi Omni-directional operation in the 2.4 Ghz band. It extends the signal coverage[1] and delivers constant wireless stability in all directions. The wireless adapter bridges the exchange of navigation feedback and operator commands between the UGV and the GCS. The key performance specifications of the P3at are summarized in Table 6.

Table 6. Key Specifications of the P3at.

Parameter	**Specification**
Dimension (L x W x H)	508 x 497 x 277 mm
System Weight w/o peripherals	< 12 kg
Maximum Speed	0.7 m/s
Continuous Endurance Time w/o load	< 4 hours
Processor	Intel i7-3770
RAM Speed	16 GB
INU Dimension	44.2 x 36.6 x 11 mm
INU Weight	< 20 g
INU Position Output Rate (EKF INU & GPS)	1 to 4 Hz
External Wireless Antenna Gain	8 dBi

[1] The signal coverage is affected by the EIRP: EIRP = Power $_{\text{Transmitter}}$ + Gain $_{\text{Antenna}}$ – Loss $_{\text{Cable}}$

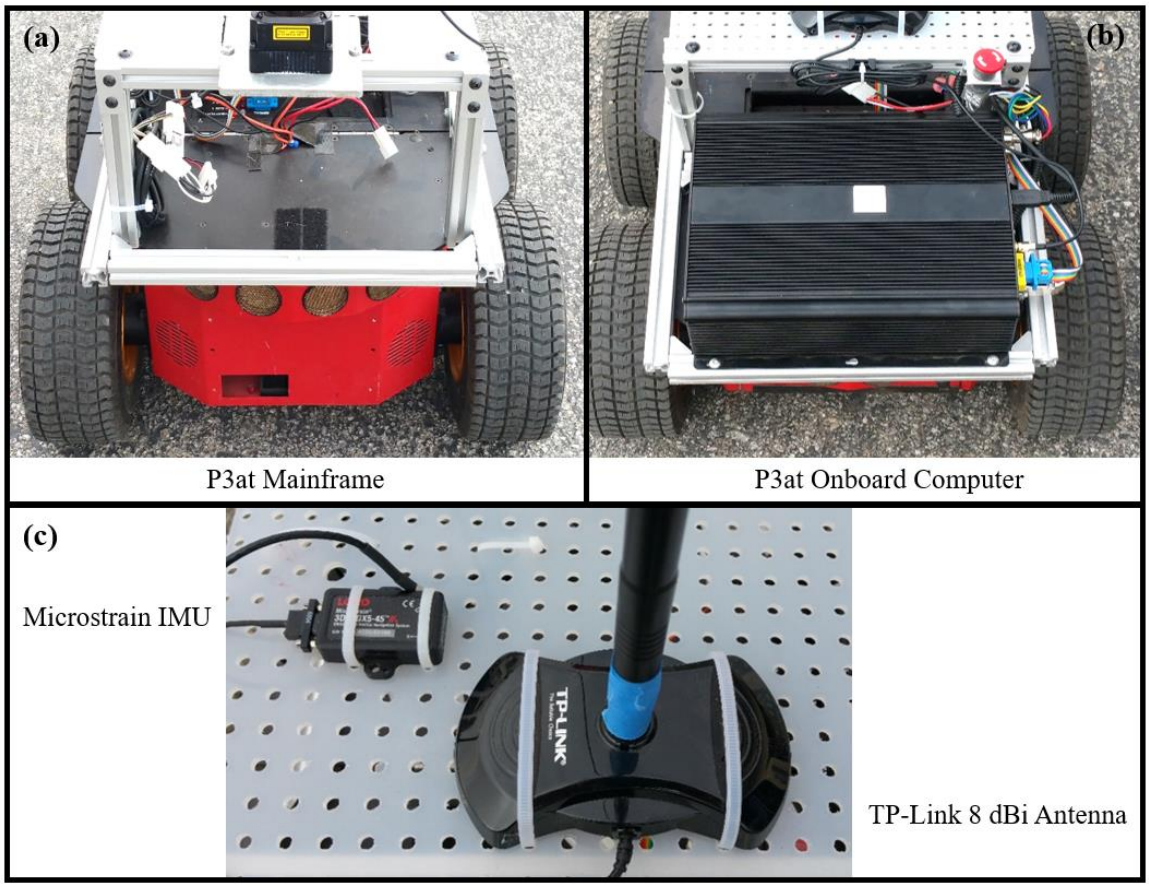


Figure 21. Components of the P3at.

b. Supporting Software

Similar to the Matrice 100 OES, the Ubuntu 14.04 is installed as the OS for the basis of other programs. MATLAB version 2016a forms the main software architecture for extracting the data from the various sensors and then processing them into control commands. Residing within MATLAB are the required toolboxes and libraries used to interface with other software and attached devices. MATLAB also has the processing capabilities to compute large metadata and handle numerous concurrent running threads.

ROS Indigo version is employed to connect the P3ats sensors and actuators, as well as the software architecture. It manages the transfer of packages and commands from the MATLAB scripts to the P3ats, executing the terminal control of the UGVs. ROS bridges the communications among the various P3ats devices, such as the INU and GPS, and feeds back the data for downstream processing to facilitate the execution of navigation instructions.

4. Ground Control Station

The GCS forms the bedrock for the communication network of the overall system architecture. The GCS consists of two main components, the mainframe computer and the wireless router, as shown in Figure 22.

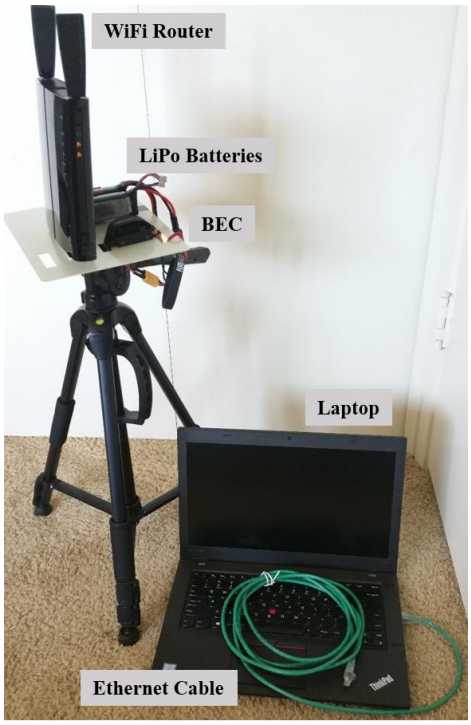


Figure 22. Pictorial Overview of GCS.

a. Hardware Specifications and Configurations

The main computer, a Lenovo ThinkPad T460, is used by the operator to control both the Matrice 100 and P3ats OES. This computer utilizes an Intel Core i5-6200U Processor operating at 2.3 Ghz, which is sufficient to handle the process load. The T460, with up to 18 hours of endurance time, features Power Bridge Technology that combines an internal battery with an external swappable battery, providing the flexibility to change batteries without powering down. Connected to the local area network (LAN), the laptop can communicate with other connected devices, such as the UAV and UGVs OES. The T460 is shown in Figure 23a.

The Buffalo AirStation N600 router, as depicted in Figure 23b, is used to support the entire communication network for the overall system. It bridges the communication among all the LAN connected devices, allowing the exchange of data, feedback, and commands. The simultaneous dual band router, providing wireless operation on both 2.4 Ghz and 5 Ghz, is backward compatible with 802.11 a/b/g WiFi protocols. The high-speed transmission (up to 1000 Mbps) utilizes Direct Sequence Spread Spectrum (DSSS).

The GCS is deigned to be portable and deployable in the field. Its power solution, as shown in Figure 23c, is provided by attached LiPo batteries. Two 2S (7.4V each) LiPo batteries are connected in a series configuration providing 14.8V 2000 mAh power to the router. A Castle Creations BEC Pro is used to step down the high 22 V to the required 12 V input for the router. The key performance specifications of the platform are summarized in Table 7.

Table 7. Key Specifications of the GCS. Adapted from Lenovo (2017); Buffalo (2017).

Parameter	**Specification**
Thinkpad T460 Dimensions (L x W x H)	339 x 232.5 x 21 mm
Thinkpad T460 Weight	< 2 Kg
T460 Endurance on Portable Battery	< 18 hours, internal battery (23 W/h), external battery (73 W/h))
T460 Processor	Intel Core i5-6200U @ 2.3 Ghz
N600 Router Dimensions (L x W x H)	16.5 x 15.7 x 3.6 mm
N600 Weight	< 335 grams
N600 Power Consumption	13.2 W max @ 12 V
Two Serially-Connected LiPo Batteries	14.8 V, 2000 mAH, 10 C
CC BEC Pro Adjustable Output Voltage	4.8 to 12.5 V
CC BEC Pro Peak Current	20 A

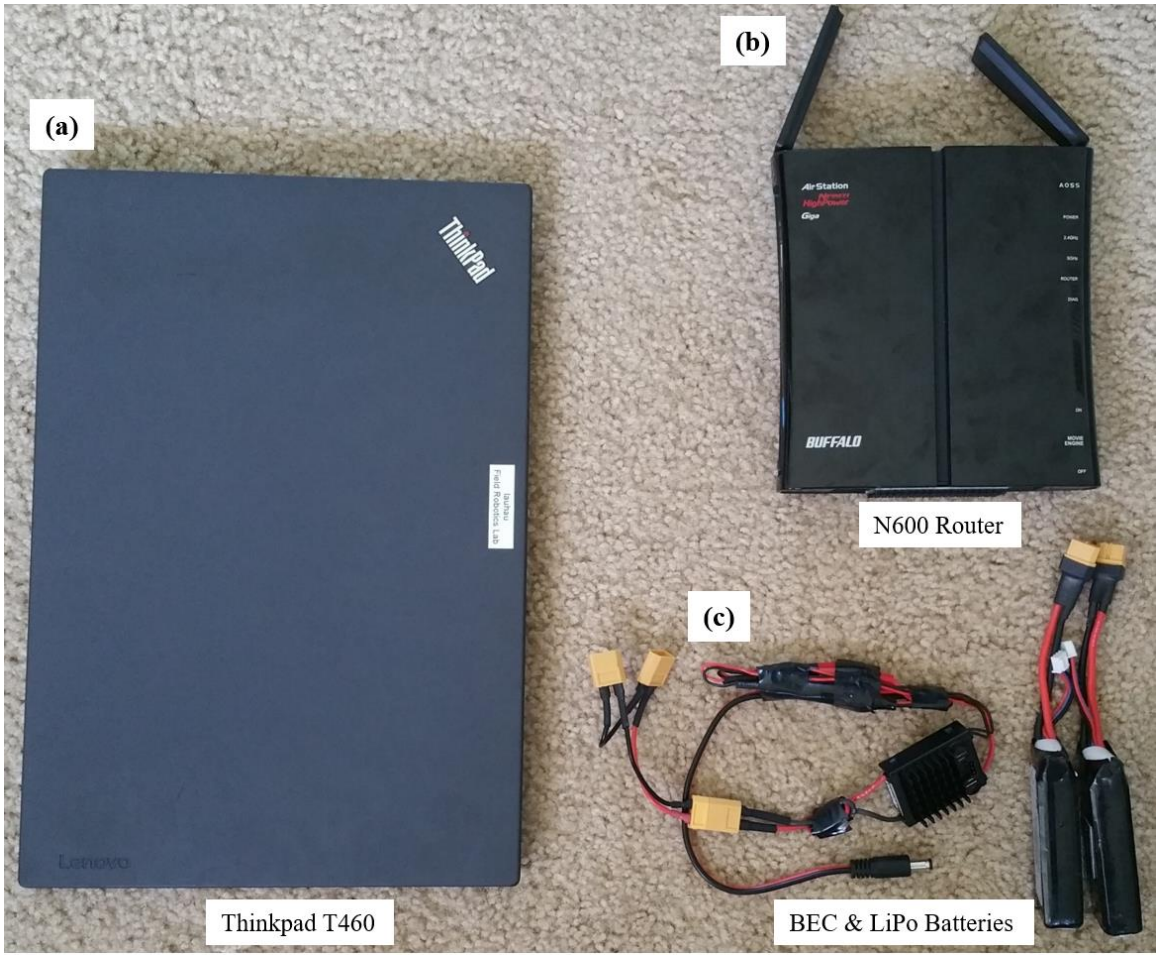


Figure 23. Components of the GCS.

b. Supporting Software

The mainframe computer uses the exact same software as that of the Matrice 100 OES and the P3ats. The installed software includes the Ubuntu 14.04 OS, MATLAB version 2016a, and Python 2.7 scripting language, ROS middleware, and other Linux peripheral software add-ons.

The N600 router is pre-flashed with the open source DD-WRT firmware, allowing the router to support a range of functionalities and configurations. Static Dynamic Host Configuration Protocol (DHCP) is enabled using the DD-WRT firmware. Each connected device, tagged with its own unique MAC address, is assigned a static

Internet Protocol (IP) address. This allows for ease of connectivity and prevents conflicts in the assignment of IP addresses.

Castle Link Programming Installer with firmware version of 3.72 is utilized to program the CC BEC Pro to regulate the high voltage input to the required voltage output of 12 V for the router. Appendix C describes the instructions for establishing the LAN and connections among the devices.

F. COMPONENT EVALUATION

It is important to ensure the components fulfill the system functional requirements identified in SV-4a. This can be achieved by analyzing the various components and conducting functional mapping. The components are also evaluated for trade-offs to ascertain the optimal architecture for the research.

1. Functional Traceability

Functional traceability was conducted to map the relationships between the functions and the objects. To this end, each object must accomplish at least one function, and each function must be assigned to a physical object. This ensures that the system architecture is able to fulfill the required system functions. Functional traceability also identifies unnecessary redundancy and critical structural components. Table 8 presents the traceability matrix.

Table 8. Functional Traceability Matrix.

System Functions			Components												
			Batteries & PDB	Flight Controller (N1)	Flight Actuators	Zenmuse Camera	Matrice 100 Ports & Cables	Matrice 100 antennas	Remote Control & Go App	Odroid-XU4	Shifter Shield	Visual Camera (C920)	Odroid WiFi Adapter	P3at	GCS
1.0		**Power Up System**	x											*x*	
	1.1	Provide Power to UAV	x												
	1.2	Provide Power to Onboard Processor	x								x				
	1.3	Conduct Built-In Tests		x											
	1.4	Transmit Telemetry		x				x	x						x
2.0		**Maneuver in Flight**			x									*x*	
	2.1	Provide Thrust & Propulsion			x										
	2.2	Provide Steering Kinematics			x				x						
3.0		**Conduct Flight Control**		x										*x*	
	3.1	Determine GPS and Compass Bearing		x											
	3.2	Accept Remote Control Input					x		x						x
	3.3	Execute Autonomous Navigation		x	x		x								x
4.0		**Detect & Track Autonomously**				x				x		x			
	4.1	Estimate Background Motion								x					
	4.2	Determine & Prune Moving Objects								x		x			
	4.3	Cluster and Track Targets								x					
5.0		**Communicate Externally**						x					x	*x*	x
	5.1	Receive Wireless Signals						x					x		x
	5.2	Process Wireless Signals						x							x
	5.3	Transmit Wireless Signals						x					x		x

2. Trade-Off Analysis

From the functional traceability, the GCS is identified as the central critical component that supports and interfaces with most of the system architecture. It is capable of utilizing different WiFi protocols to build the LAN. The various protocols range across IEEE 802.11 a/b/g/n/ac, but can be distilled into two main categories based on operating frequency: 2.4 Ghz or 5.8 Ghz. It is imperative to select and implement a suitable network communication protocol to meet the system requirements.

The Pugh process is used as the decision support method to select the optimal WiFi operating frequency, 2.4 Ghz or 5.8 Ghz, to build the network. The process first

identifies the various criteria used to evaluate the alternatives. Second, it benchmarks the alternatives against a datum such as the current system. Each alternative is evaluated as either better (+ or 1), same (S or 0), or worse (- or -1). Lastly, the Pugh process requires the computation of an overall score to determine the optimal alternative (Lønmo and Gerrit 2014).

The selection of alternatives is based on three principle considerations that form the evaluation criteria for the Pugh process. The three evaluation criteria are detailed in the following paragraphs.

a. Link Quality

The link quality affects the connection and the subsequent transmission efficiency of the connections. Link quality is governed by the signal-to-noise ratio, which is typically measured in decibels (dB). Link quality with a higher dB will have more stable and better transmission rates that lower dB link quality. Link quality is affected by numerous factors, with the main causal factor being the channel/ frequency quality. Interference in a particular channel or frequency will have a negative impact on the link quality causing packet losses or frequent disconnections. The more networks operating on the same channel or frequency will result in more interference. Attenuation in signal transmission is caused by variations in the transmission medium, such as haze or multi-pathing effect amplified in urban areas, which also affect the link quality negatively.

b. Bandwidth

To achieve near real-time detection and tracking of multiple moving objects, operators require the transmission of annotated video feeds. Video feeds are inherently large data files that require high-speed data transmission to deliver applicable and usable effects. The bandwidth of the transmission determines the amount of data transmitted per unit time. Thus, relatively higher bandwidth will result in speedier transfer rates.

c. Range

Unmanned systems are commonly used to perform dull, dirty, and dangerous missions. They are usually operated at a distance to ensure the safety and reduced

footprint of the operators. The benefits of autonomous detection and tracking of moving objects are optimized when the unmanned systems are operated at a distance from the operators. Moving objects in proximity are obvious to the operators. Thus, stable connections capable of extended range are generally preferred.

d. Evaluation Results

The Pugh process was conducted based on the previously mentioned three evaluation criteria. The alternatives of using 2.4 Gz and 5.8 Ghz were then benchmarked against the datum of using 2.4 Ghz to provide the communication network. Table 9 summarizes the evaluation of the two alternatives.

Table 9. Pugh Matrix for Selecting WiFi Operating Range.

Criteria	**Alternatives**	
	2.4 Ghz	*5.8 Ghz*
Link Quality	DATUM	+
Bandwidth		+
Range		-
Overall Score	0	1

The 5.8 Ghz option is evaluated as the optimal WiFi operating frequency to support the overall system architecture. This protocol boasts a better link quality over the 2.4 Ghz frequency as it is relatively less crowded. Furthermore, 2.4 Ghz is widely used by household devices such as microwaves and connectivity devices, resulting in an overcrowded frequency domain. In addition, 5.8 Ghz with higher frequency, has more channels and lesser frequency overlaps than 2.4 Ghz. This results in less interference for the 5.8 Ghz frequency. With higher frequencies, 5.8 Ghz have higher bandwidth for faster data transmission than 2.4 Ghz. However, higher frequencies have lower transmission range with weaker penetration power. Although 5.8 Ghz is evaluated as the

suitable alternative for this particular system architecture, varying operator needs might call for a different operating frequency.

3. Communication Links

Now that the communication mode has been evaluated and selected, the proposed system architecture illustrated by DODAF SV-1 can be expanded with the depiction of the communication links and networks. DODAF System View-2 (SV-2), as shown in Figure 24, details how the interfaces between system nodes are implemented.

The inter-nodal communication linkages among the system nodes are connected by the WiFi transmission protocol in the 2.4 Ghz or the 5.8 Ghz ISM band. Operators utilize the GCS network to connect with the UAV and UGVs OES via these WiFi signals. As detailed in the trade-off analysis, the selection of the 2.4 or 5.8 Ghz operating frequency depends on the users' requirement. The operators gain access to the various OES by the Secure Shell connection. The subsequent control commands are issued either by a BASH terminal, Python, or MATLAB commands.

The intra-nodal communication linkages within the system nodes are governed primarily via TTL or proprietary communication protocols such as the FHSS protocol used by the Matrice 100 and its remote control.

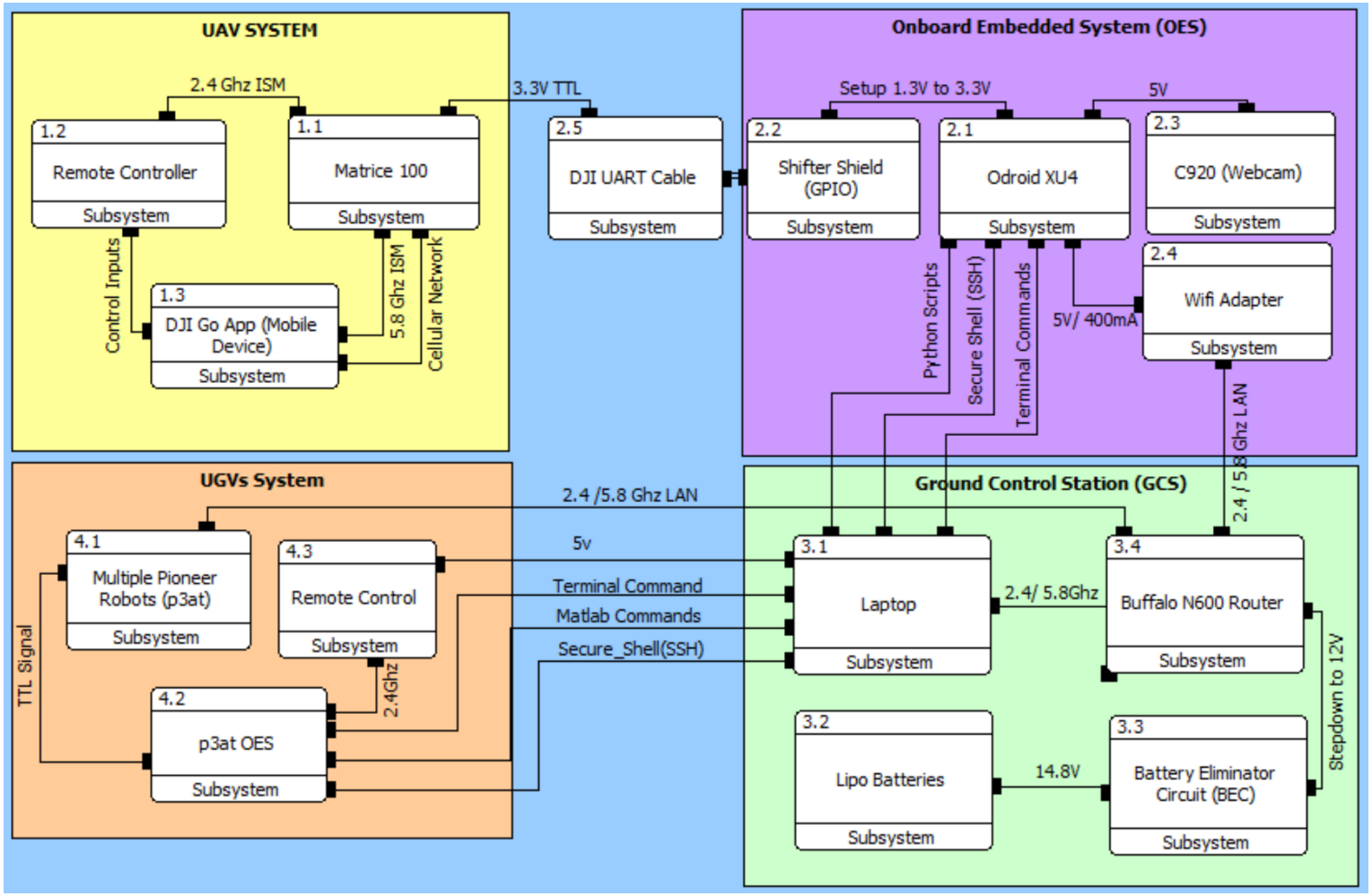


Figure 24. SV-2 of System Physical Architecture.

III. ALGORITHMIC DESIGN

A. DETECTION AND TRACKING OF MULTIPLE MOVING TARGETS

This thesis research is part of a joint collaboration between Purdue University and the Naval Postgraduate School (NPS). The multi-target detection and tracking algorithm was developed by Purdue University. NPS integrated and implemented the algorithm in the proposed system for experimental tests and evaluation.

Several different camera-based approaches exist to support the detection and tracking of moving objects. Commonly used methods involve machine learning techniques, such as Convolutional Neural Networks (Serre et al. 2007) and Random Forests (Bosch et al. 2007), which recognize the appearance of the target objects using training datasets. Machine-learning techniques are proven to be effective in identifying objects in complex environments with background clutter and varying illumination. Nonetheless, the techniques generally only work on relatively large and visible objects with distinct features. In addition, machine-learning techniques are relatively computationally intensive for near real-time implementation (Li et al. 2016).

Other computer vision methods involve the use of motion information from the moving targets. Such motion-based approaches mainly involve background subtraction and optical flow techniques. Background subtraction techniques remove the pixels with the same brightness constancy over time, thus revealing the moving objects (Pentland et al. 1999). However, this technique is not reliable in fast moving situations where background is not easily compensated. Another approach, the optical flow technique, detects moving targets by identifying their local motion vectors through the examination of the various sequential frames (Brox and Malik 2011). While the optical flow technique is computationally less intensive, its accuracy of detection depends heavily on the quality of the pixels local motion vectors. Distorted images with poor image quality reduce the overall accuracy.

The CV algorithm used in this research utilizes the motion-based approach, combining the background subtraction and optical flow based methods to detect and track

multiple moving targets. Figure 25 summarizes the main components of the Purdue CV algorithm while the following sections detail the mechanisms of the various components.

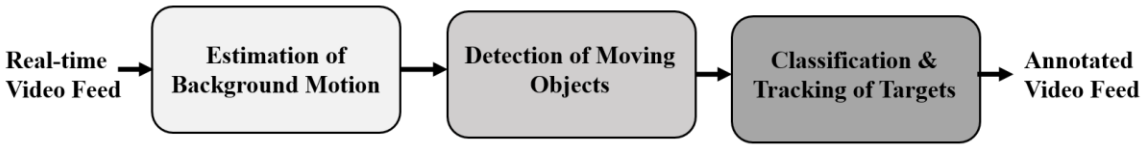


Figure 25. Overview of Purdue CV Algorithm.

1. Estimation of Background Motion

Background motion estimation is performed on a set of identified salient points over consecutive frames. The local motion fields of the salient points are approximated and subsequently fitted into a global transformation that will represent the overall background motion of the UAV (Li et al. 2016). Figure 26 illustrates the steps performed to estimate the background motion.

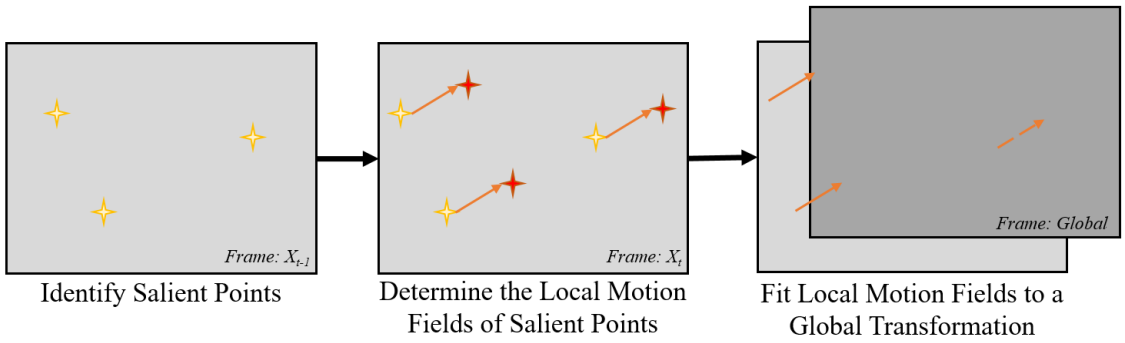


Figure 26. Process for Estimating Background Motion.

First, to extract the salient points, Shi-Tomasi Corner detection is used to determine the persistent points over several frames by assigning thresholds on the saliency value Q. λ_1 and λ_2 are the eigenvalues of the precision matrix computed in the local autocorrelation function. The salient points are selected to ensure spatial distribution across the entire frame:

$$Q(s) = \min\{\lambda_1, \lambda_2\} \tag{4}$$

Second, the local motion vector, u_t, is computed by examining the identified salient points $p_t - 1$ over consecutive frames X_t and X_{t-1}. Since the neighboring points $N(p_t - 1)$ should have similar motion (Lucas and Kanade 1981), the computation is solved for the least square problem. Each pixel is denoted as s.

$$u_t = \arg\min \sum_{N(p_t - 1)} |X_t(s+u) - X_{t-1}(s)|^2 \tag{5}$$

Finally, the respective local motion fields of the set of corresponding points P_t and P_{t-1} in X_t and X_{t-1}, are fitted into the global transformation H_t using the perspective transformation model (Li et al. 2016):

$$H_t = \arg\min \sum_{p_t \in P_t, p_{t-1} \in P_{t-1}} \| p_t - H \circ p_{t-1} \|_2^2 \tag{6}$$

2. Detection of Moving Objects

Background motion is subtracted from the frames to isolate moving objects that are assumed to have more complex motion than the background. Salient points on the moving objects are identified and their local motion fields are computed. These salient points are pruned based on the motion difference threshold to remove any noise (Li et al. 2016). Figure 27 summarized the process of detection.

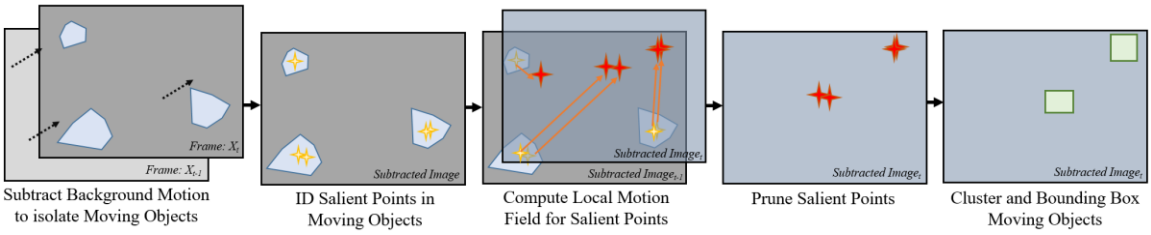


Figure 27. Process for Detecting Moving Objects.

First, the estimated background motion of multiple previous frames is subtracted from the original image to compute the background subtracted image E_{t-1} of frame X_{t-1}. This is achieved by taking the average of forward and backward tracing:

$$E_{t-1} = \frac{1}{2} | X_{t-1} - H_{t-1} \circ X_{t-2} | + \frac{1}{2} | X_{t-1} - (H_t)^{-1} \circ X_t | \tag{7}$$

Second, as explained in section A1 of this chapter, Shi-Tomasi Corner detection is used on E_{t-1} to extract the salient points q_{t-1}. The Lucas-Kanade method is then applied to determine the local motion field v_t:

$$v_t = \arg\min \sum_{N(q_t - 1)} |X_t(s+v) - X_{t-1}(s)|^2 \tag{8}$$

Finally, based on the assumption that moving targets have different motion from the background, the salient points are pruned according to the difference between the estimated background h_t and the local motion field v_t. The pruned points r_t are determined based on the empirical threshold T of motion difference d_t:

$$d_t = h_t - v_t \tag{9}$$

$$r_t = q_{t-1} + v_t \qquad if \ \| d_t \|_2 > T \tag{10}$$

The pruned points are clustered as a single moving object according to their spatial proximity by applying connected component labeling (Samet and Tamminen 1988). Bounding boxes are then generated for each moving object.

3. Classification and Tracking of Targets

To enhance the accuracy of detection, Li et al. (2016) pitted the spatio-temporal features of the clustered objects against empirical thresholds. The tracking technique is then applied to classified targets to ensure coherent temporal signatures, thereby reducing intermittent misdetections and false alarms. Figure 28 illustrates the classification process.

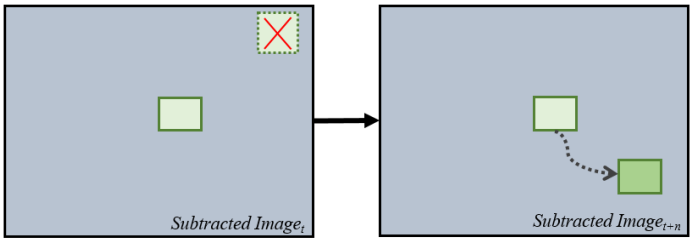


Figure 28. Process for Classifying and Tracking Targets.

The classification of moving targets is based on two assumptions. The first assumption is that the target is non-deformable, and thus, its angle variance f_t in the motion field vectors D_t is consistent. S_t represents the number of points in the clustered object:

$$f_t = \frac{\sum_{D_t} |\arctan d_t - \mu_\theta|^2}{S_t}$$

where (11)

$$\mu_\theta = \frac{\sum_{D_t} \arctan d_t}{S_t}$$

The second assumption is that the target has densely distributed points density g_t within the enclosing bounding box B_t:

$$g_t = \frac{S_t}{B_t} \tag{12}$$

A classifier label y_t is generated with empirical thresholds T_1 and T_2 for angle variance and point density, respectively. A positive value indicates that the moving object is indeed a target.

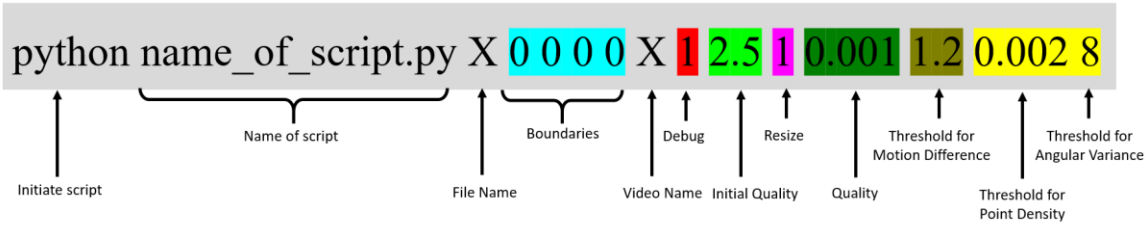
$$y_t = 1, \text{ if } f_t < T_1 \text{ and } g_t > T_2 \tag{13}$$

$$y_t = 0, \text{ otherwise}$$

Kalman filtering is then applied to track and enforce coherent temporal signatures of the classified targets (Welch and Bishop 2006). This reduces the intermittent misdetections and false alarms.

4. Layout of CV Algorithm Command

At this juncture in the research, the CV algorithm is still in the midst of continual revision and regular updates. The algorithm encompasses the aforementioned series of image processes in python scripts. In light of human systems integration, the Python command is coded to allow easy tuning of the CV algorithm parameters by the users without the need to access the scripts during operations. Figure 29 illustrates the layout of the CV algorithm command and lists the description of the adjustable input parameters.



Boundaries	: To mask off area of the video feed if constant features are in view (e.g.) propellers or UAV body.
Debug	: Binary control to toggle between screening of live annotated video feedback or running in the background.
Initial Quality	: The quality level of feature points used to compute perspective transform.
Resize	: Use to proportionally resize or down sample the resolution of video feed.
Quality	: The quality level of feature points used to compute Background Motion Estimation.
Motion Difference	: Threshold to prune off small motion differences.
Point Density	: Classifier Parameter T_2 used to threshold to prune off sparse cluster.
Angular Variance	: Classifier Parameter T_1 used to threshold to prune off large variance in motion.

Figure 29. Decomposition of CV Algorithm Command.

B. NAVIGATION CONTROLS OF UGVs

UGVs simulate moving targets for the testing of the CV algorithm. Because the system architecture is designed to require minimal operator inputs, the UGVs can be controlled manually via remote control or programmed to perform autonomous waypoint navigation. This section describes the technical design enabling autonomous navigation of the UGVs. The instructions for setting up and manually controlling the UGVs appear in Appendix D.

Multiple UGVs can be controlled simultaneously via the use of a leader and follower hierarchy. One UGV can be appointed as the leader, while the following UGVs will use the leader's global positions as their waypoint goals. To achieve autonomous waypoint navigation, the operator and UGVs are required to perform a series of actions as illustrated in Figure 30.

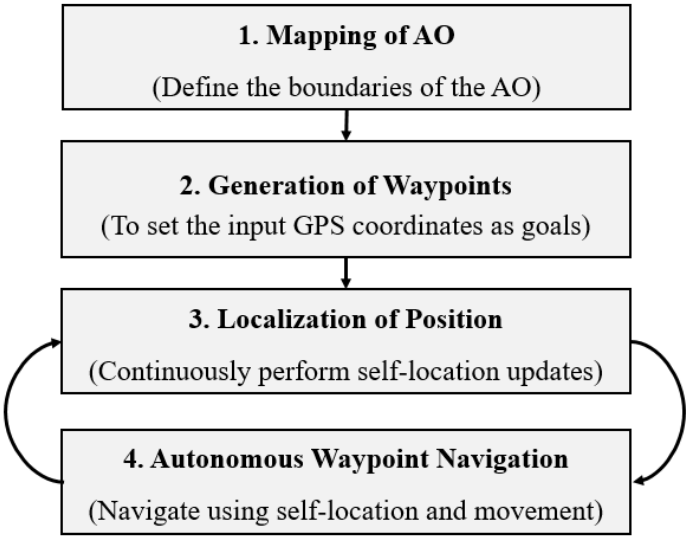


Figure 30. Flow Diagram for Autonomous Waypoint Navigation.

First, the operator needs to map the entire AO to define the boundaries in which the UGVs are operating. This is achieved manually via remote control. Such mapping of the AO mimics real humanitarian aid/disaster relief missions where forward reconnaissance elements are inserted into the AO to collect information on the terrain and situation. Second, from the mapped AO, the operator then inputs the GPS coordinates of the waypoints and runs the navigation program. The UGVs use these pre-defined waypoints as their navigation goals.

Subsequently, the UGVs conduct an iterative process of self-localization and autonomous navigation. Self-localization is based on the data from the INU. It coagulates the inputs from INU sensors such as wheel odometry and the GPS sensor using Kalman

filtering. This produces the location of the UGVs in reference to their global frame position. Autonomous waypoint navigation can then be achieved.

The autonomous waypoint navigation function consists of a main script that calls on two sub-scripts using the call-back function. The call-back function reduces the computational time and provides simultaneous threading. Figure 31 shows the pseudocode for the main script used. The scripts used are adapted from the NPS multi-robot control classes conducted by Bingham (2017).

Input of Waypoint Coordinates
- Input the desired GPS coordinates
- Passes the waypoints as goals for robot navigation

Self-Localization of Robot:
- Subscribe to the input from the IMU

Perform Navigation:
- Call-back the Waypoint Navigation script
  If robot navigate to goal: Done
  Else: Keep running Waypoint Navigation script

Figure 31. Pseudocode of Autonomous Navigation Main Script.

The main script calls on the use of waypoint navigation script, which is shown in Figure 32. The loop implementation ensures that the UGVs cycle through all the waypoints and they navigate to a desired proximity for each waypoint. This radial proximity is based on an empirical threshold. The waypoint navigation script calls on the control script. The control of the waypoint navigation can be achieved by either the waypoint control algorithm or LOS path following algorithm.

Determining current pose:
- Read in data from INU sensors.

Perform Waypoint Navigation based on current pose:
- Call-back Waypoint Control / LOS Path Following script

Check Self-Localization:
- If distance of Robot current pose to waypoint is less than threshold:
  If next waypoint exists: move to next waypoint
  Else: Stop Robot, Navigation Completed
- Else: Move Robot as per Waypoint Navigation

Figure 32. Pseudocode of Waypoint Navigation Script.

1. Waypoint Control

The waypoint control algorithm is used to guide the UGVs from their current position toward the desired waypoints. This algorithm is simple to implement with short computational time. Figure 33 depicts the waypoint control concept.

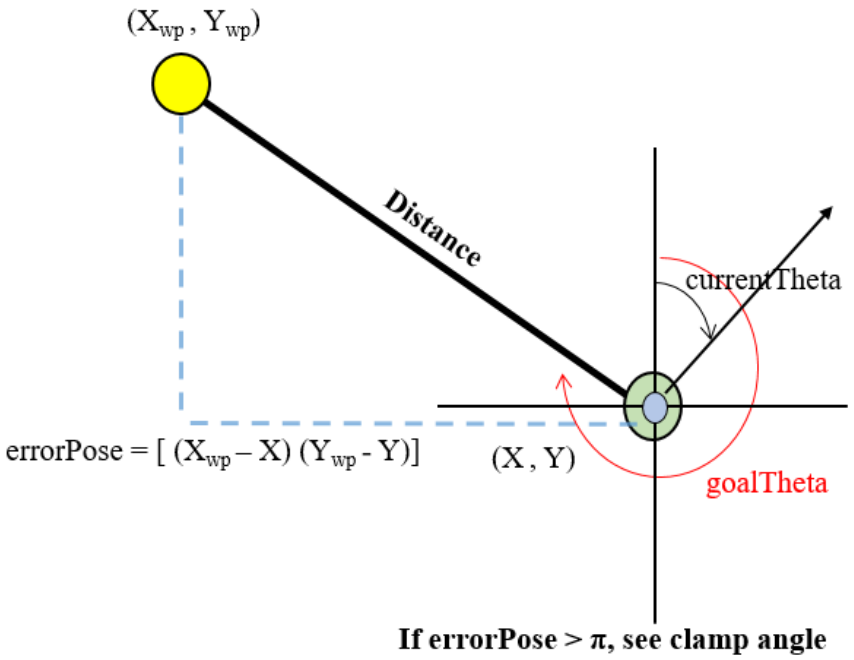


Figure 33. Concept of Waypoint Control.

The objective of navigating from the current position to the desired waypoint is achieved by computing the distance via Pythagorean Theorem, the required bearing via trigonometry and applying the angular and linear gains for movement. Figure 34 lists the pseudocode for the waypoint control.

Set the tunable parameters:
- angular gain, linear gain

Determine the distance (hypotenuse) from current position to waypoint
- Compute errorPose

Determine the difference in bearing angle between current heading and waypoint bearing
- Compute errorTheta

Determine the optimal bearing to turn based on the errorTheta
- Apply clamp angle to optimise the direction of turning for the robot

Apply angular and linear gains for movement

Figure 34. Pseudocode for Waypoint Control Algorithm.

Clamp angle is applied to optimize the bearing direction based on the UGVs' current heading, as shown in Figure 35. The magnitude of the *errorTheta*, the difference between UGV current heading and waypoint bearing, determines the direction that the UGVs will turn; clockwise (CW) or counter-clock wise (CCW).

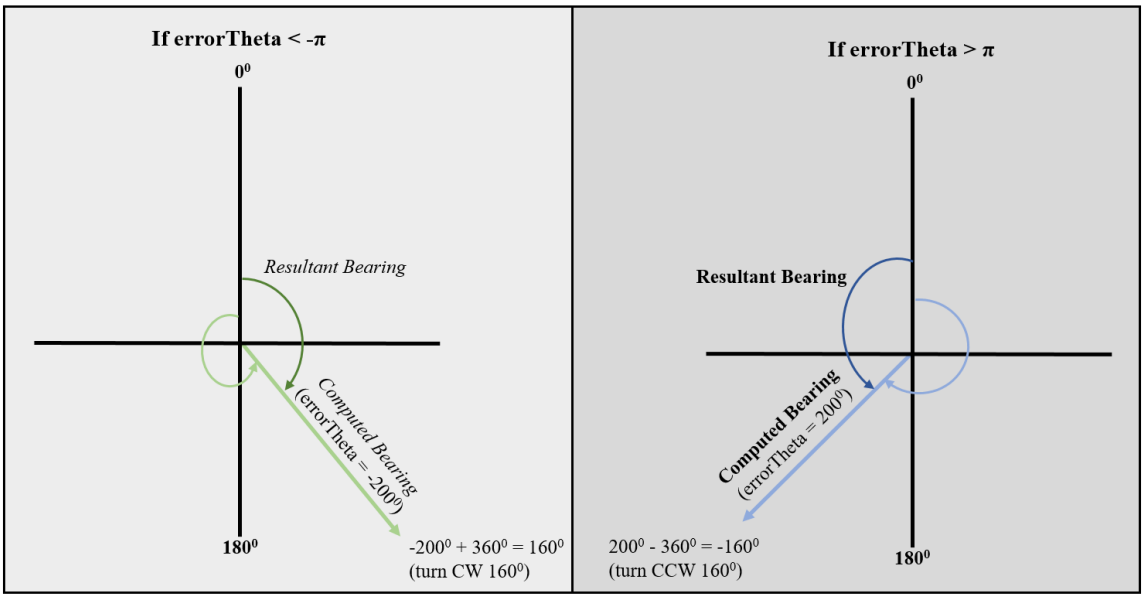


Figure 35. Clamp Angle for Optimizing Bearing.

Although waypoint control navigates the UGVs to the waypoints, it is prone to cross-track error. Cross-track error, as illustrated in Figure 36, is the veering off-course by the UGVs from the intended path of travel. This results in an unpredictable navigation pattern and unnecessary distance and time penalty.

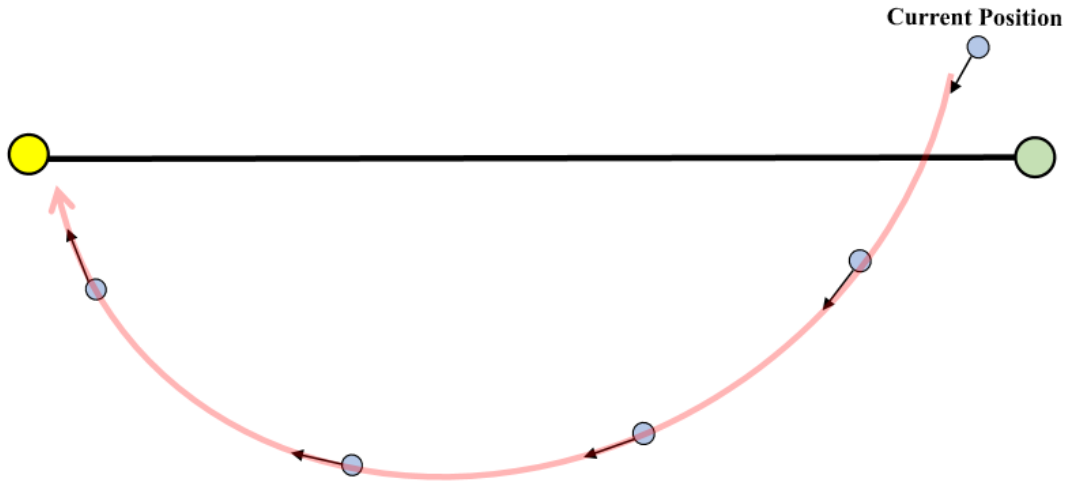


Figure 36. Cross-Track Error by the UGV.

2. LOS Path Following

The LOS path following algorithm is used to guide the UGV in a relatively straight linear path toward the navigation goal. This algorithm reduces erroneous navigation by reducing cross-track error while navigating toward the goals (Bingham 2017). However, LOS path following algorithm can be computationally intensive, resulting in longer reaction time by the UGV. Figure 37 illustrates the LOS path following approach.

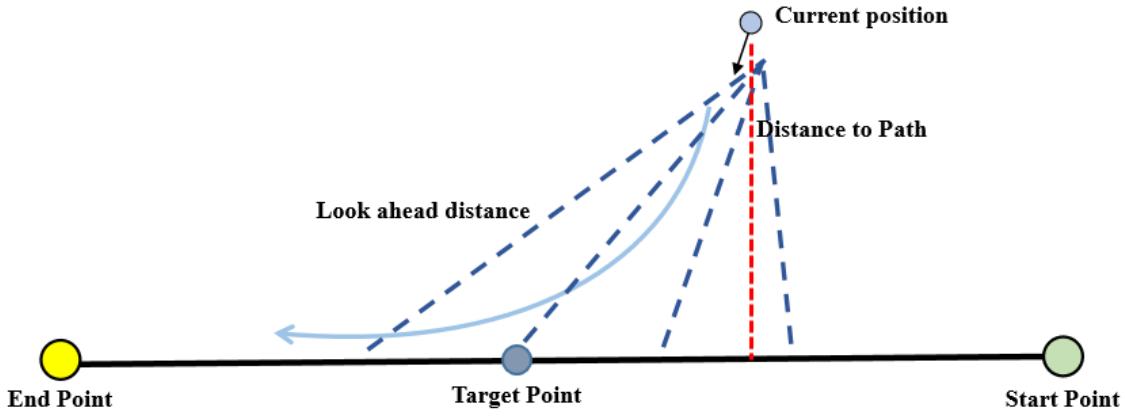


Figure 37. Concept of LOS Path Following.

The objective is for a UGV at its current position to move from its start point toward its end goal in an intended straight-line path. This is achieved by determining the distance from the UGV to the navigation path, selecting a target on the navigation path, and applying the angular and linear velocities gains for the movement (Bingham 2017). Figure 38 shows the pseudocode for the LOS path following algorithm.

Set the tunable parameters:

- Look_ahead_distance, angular gain, linear gain

Determine the gradient of the navigation line from start point to end point

- Set conditions for vertical, horizontal and slope

Determine the intersection point between the robot and the navigation line (X_{int} Y_{int}):

- Use the gradient formula Y = mX + c

Find the a target along the navigation line which will guide the movement of the robot to converge to the path:

- Use look_ahead_distance to pick a target point
- Check if target point is beyond the end or start point

Determine the heading angle and velocities:

- Apply clamp angle to optimize the direction of turning for the robot
- Apply angular and linear gains

Figure 38. Pseudocode for LOS Path Following Algorithm.

The angular and linear gains affect the speed and time taken for the UGVs to navigate to their goals. Excessive gain parameters cause the UGVs to overshoot their desired position, thereby leading to over-correction depicted by the wiggly navigation illustrated in Figure 39. While lower gain values produce a higher fidelity control, they incur longer computational time. The tradeoff between fidelity and speed of navigation depends on stakeholders' requirements and mission profile.

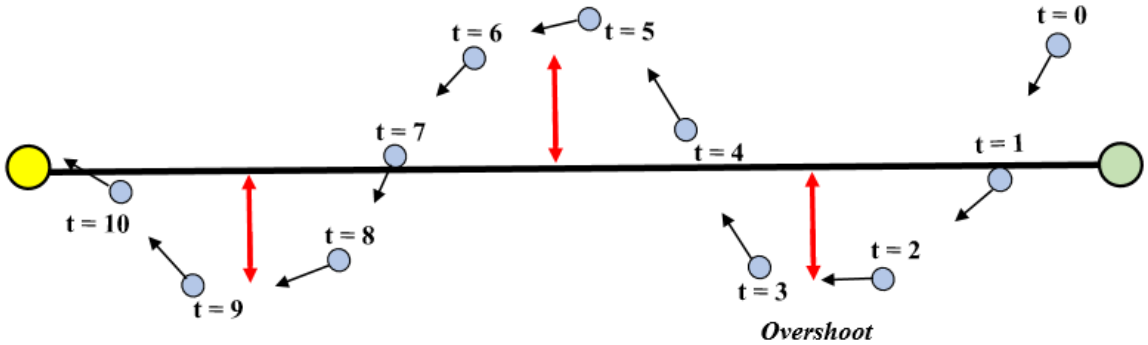


Figure 39. Overshoot and Overcorrection by High Gains.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. SIMULATION AND RESULTS

The intent of the experiments described in this chapter is to evaluate the overall system architecture in performing the autonomous detection and tracking of multiple moving targets. This proof of concept is to demonstrate the use of the CV algorithm to enhance situation awareness in a crowded AO. Design considerations were implemented to ensure that the experiments could be conducted by a minimal number of operators, ranging from one to two users, despite the use of multiple unmanned systems. The moving targets simulated by the UGVs have been programmed for autonomous navigation, allowing the operator to focus on operating the UAV loaded with the CV algorithm.

The operation of the UAVs was sanctioned by NPS Aviation Operations and NPS ACOS-Aviation Activities, and the respective UAV systems have been individually cleared. Prior to each takeoff, the Air Boss of the airfield was notified to coordinate the use of the air space. All flight operations were made in adherence with FAA rules and regulations, for which each flight's activities were logged and submitted.

A. INITIAL EXPERIMENT SETUP

Prior to the conduct of experiment, the operators have to consider and set up the experimental conditions. This involves conducting a thorough check of the overall system architecture to ensure the safety of operation and exploring the various aspects of the system to achieve a holistic experimental design.

1. Pre-operation Checklist

The conduct of experiment involves simultaneous operation of numerous unmanned systems and the use of several different interfaces. Although the overall system architecture is designed for a minimal number of operator inputs, it is imperative to ensure that all subsystems are functioning properly before conducting integrated system operation and evaluation. The pre-operation checklist, given in Table 10, serves as a guide to check the serviceability if the subsystem. In addition to completing the

checklist, the operator must exercise caution and ensure the safety of the surroundings before performing any activities.

Table 10. Pre-operation Checklist.

Checklist
Operation of GCS
- Ensure two LiPo batteries are fully charged
- Connect LiPo batteries to BEC and to GCS Router
- Router booted up with blinking orange indicator lights at both 2.4 and 5.8 Ghz
- Ensure WiFi connection is established to the GCS laptop
- Ensure that the GCS laptop has adequate power for the entire operation
Operation of UGVs
- Ensure that P3at UGVs are adequately charged
- Boot up P3at UGVs, LAN connection to GCS will be established automatically upon successful bootup
- SSH or ping UGVs to veify connection
- Verify ROS architecture via *rqtgraph* command
- Check UGVs sensor status such as INU and GPS via *rosnode list* command
- Ensure that the UGV emergency stop button is released
- Verify manual control operation by executing remote control launch file
- Verify autonomous waypoint navigation by executing Matlab script
- Ensure that the UGVs are operated within the allocated boundaries
Operation of UAV
- Connect mobile DJI GO App to DJI C1 remote control
- Boot up C1 remote control and verify connection to DJI GO App
- Ensure that the DJI battery is adqeuately charged
- Boot up Matrice 100, which will initiate the system BIT
- Ensure that the remote control and DJI GO App is connected to Matrice 100
- Verify gimbal camera operation by toggling the switches to control the camera position
- Ensure that the **Return to Home** function and **Altitude** is set.
- Ensure that the lost connection contigency failsafe mode is enabled
- Ensure that the system is **Ready to Go**, highlight in Green with more than 10 GPS fixes
Operation of the OES
- Ensure that the OES is connected to the BEC, and to the Matrice 100 XT60 power port
- For ROS integration, ensure UART cable is connected to OES shifter shield and Matrice 100 UART port
- Verify OES is booted up when Matrice 100 is turned on.
- LAN connection to the GCS should be established automatically upon sucessful boot up.
- SSH into OES from GCS laptop to verify the status of the OES
- Enable *screen* mode onboard OES
- Verify operation of the CV algorithm by executing the Python script

2. Parameters Exploration

The CV algorithm parameters can be tweaked to alter the sensitivity and efficiency of the detection and tracking. By tuning the parameters, operators can cater for varying usage, mission profiles, and specific targets. In this research, multiple UGVs and UAVs are used as moving targets. The UGVs' maximum speed is 0.7 m/s, and they are relatively bulky in size. In contrast, the UAV DJI Inspire 1 used in this research has a maximum speed of 20 m/s and is relatively small in size. These targets specifications, namely the slow moving speed and small size density, were taken into consideration during the parameter exploration process.

Two sets of target classifier parameters were tested during the evaluation of the overall system architecture; the thresholds for average and small targets. Catering for average targets, $T_1 = 5$ and $T_2 = 0.02$ were set for angle variance and point density feature, respectively. Catering for small targets, $T_1 = 8$ and $T_2 = 0.005$ were used. The rest of the parameters remain unchanged. If the target specifications or environment is unknown, it is recommended to set the classifier parameters on average values.

To implement the CV algorithm in real-time operation, it is important to ensure the computational efficiency, which is affected by the number of pixels to be processed. Two sets of resolutions for the real-time video input feed were tested for this research: 640 x 480p and 800 x 600p.

B. CONDUCT OF EXPERIMENTS

The experiments were conducted systematically to verify and validate the effectiveness of the CV algorithm, the integration onboard the UAV, and the autonomous navigation of the UGVs. These experiments progressed from a sanitized indoor environment to an outdoor environment with operational challenges such as background clutter and variation in illumination. The evaluation process included each developmental stage. The three main experimental setups were a) standalone test of CV algorithm, b) verification of prototype subsystem, and c) evaluation of overall system architecture.

The following sections detail the experiment conditions and results. Screenshots of the CV algorithm annotated videos were used to illustrate the experiment results. Although screenshots provide a visual depiction of the results, they do not provide the entire coverage of the footage for detection and tracking results. Refer to the video montage[2] of the annotated experiment results for better comprehension of the results.

1. Standalone Test of CV Algorithm

Purdue University conducted extensive evaluation of the CV algorithm on a recorded 1920 x 1080p HD resolution video dataset (Li et al. 2016). Prior to integration onto the UAV, the CV algorithm underwent standalone tests with real-time operation in contrast to the recorded videos. Executing the CV algorithm on real-time video feed instead of recorded videos provides feedback on the computational efficiency and sensitivity to environmental conditions.

a. Test Conditions

The CV algorithm was executed on direct video feed from the Logitech C920 visual camera to detect and track multiple moving targets. The multiple moving targets were simulated by two homemade micro-UAVs controlled manually via remote control. Each micro-UAV is no larger than 60 x 60 x 30 mm in dimension (L x W x H).

The standalone tests were conducted in both indoor and outdoor environments. The indoor environment provided a controlled and sanitized testbed for testing the CV algorithm before it was subjected to the operational outdoor environment. The indoor environment comprised a relatively clean setup; white background with no background clutter and ample lighting. The outdoor environment testing occurred in the field with moving flora and dust due to the downwash from the micro-UAVs.

[2] The video montage is available at https://youtu.be/xk-T4-QwHkY.

b. Test Results

In the indoor conditions, the CV algorithm was able to detect and track both the micro-UAVs consistently and reliably. Refer to Figure 40. The sanitized conditions with constant illumination allowed tweaking of the CV algorithm to achieve reliable results.

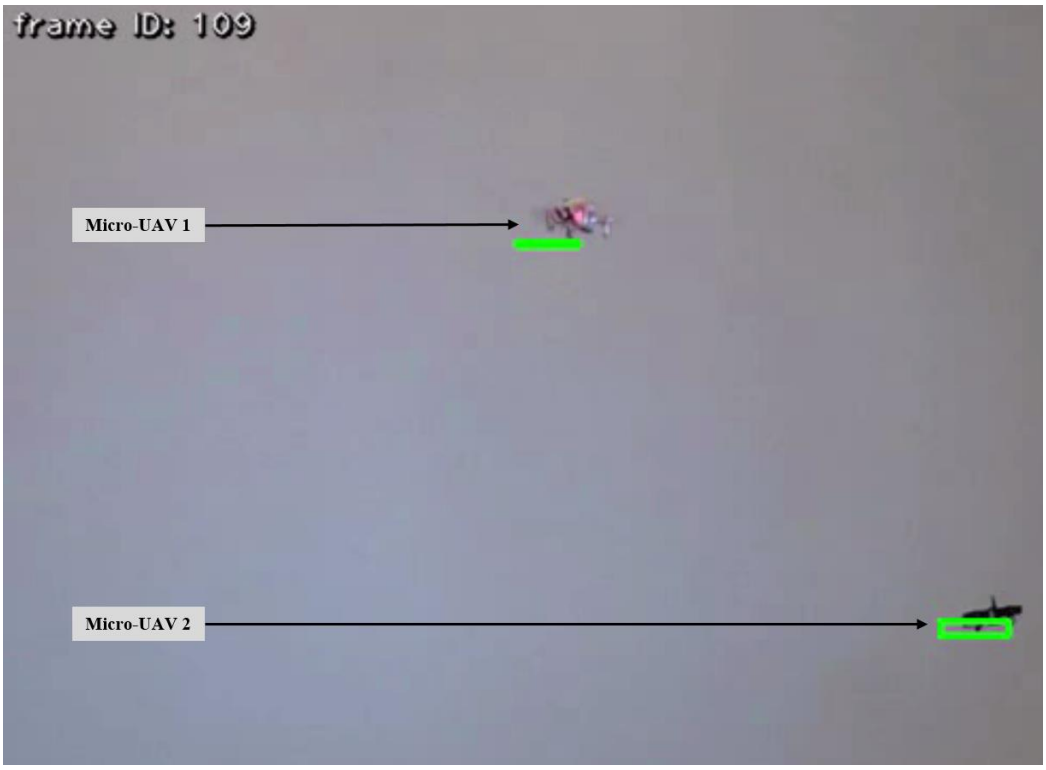


Figure 40. Screenshot of CV Algorithm in Indoor Condition.

When subjected to environmental conditions such as wind in the outdoor testing, the CV algorithm was able to capture the random flying micro-UAVs. See Figure 41 for the annotated detections. With the encouraging results from the standalone tests, the research progressed to further verify the CV algorithm for real-time operation.

Figure 41. Screenshot of CV Algorithm in Outdoor Conditions.

2. Verification of Prototype Subsystem

Upon laboratory verification, the CV algorithm was then integrated for real-time operation onboard a functional UAV. This prototype UAV subsystem must be verified before commencing the operational testing of the overall system.

a. Ground Checks

The design of the prototype subsystem was checked for any disruptive intermissions that would affect the UAV operation. The Matrice 100 automatically detects any magnetic disruptions posed by external devices, such as the OES or signal interference by other transmitters, during its BIT on bootup. Once the system verified that it was clear of intermissions, the ready to go indicator was displayed, as shown in Figure 42.

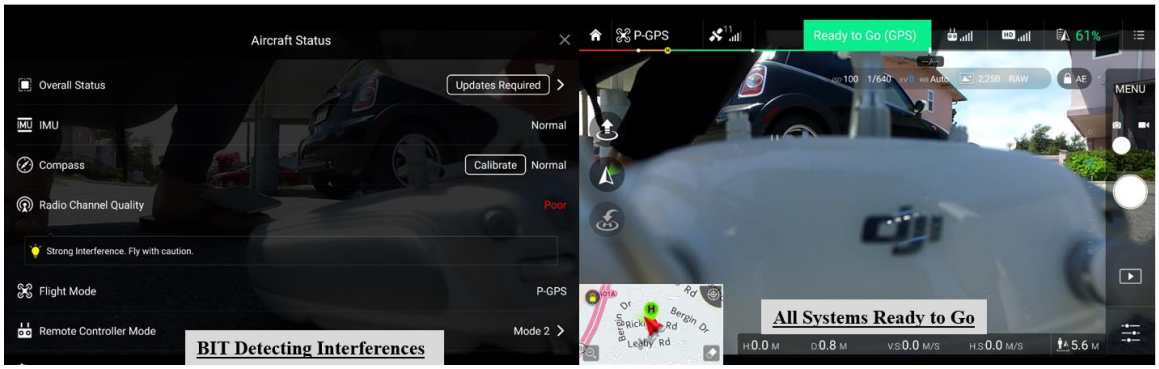


Figure 42. Screenshots of DJI Matrice 100 BIT.

Prior to the flight test, the grounded prototype subsystem was tested to ensure a feasible power solution to support the operation of the CV algorithm. An Extech 80W Switching DC Power Supply was used to quantify the amount of power used at the required voltage. Figure 43 shows the current drawn by the OES throughout the various phases of operation; namely, the boot-up of the OES, idling mode, and execution of the CV algorithm. A peak current of 3.65A was drawn for a very brief moment during the execution of the CV algorithm. Conservatively, an estimated 3 A at 5 V or 15 W of power was drawn by the OES at any instant. This translates to less than 15% of the 99.9 W supplied by the Matrice 100, leaving ample power without causing a major reduction in actual flight time.

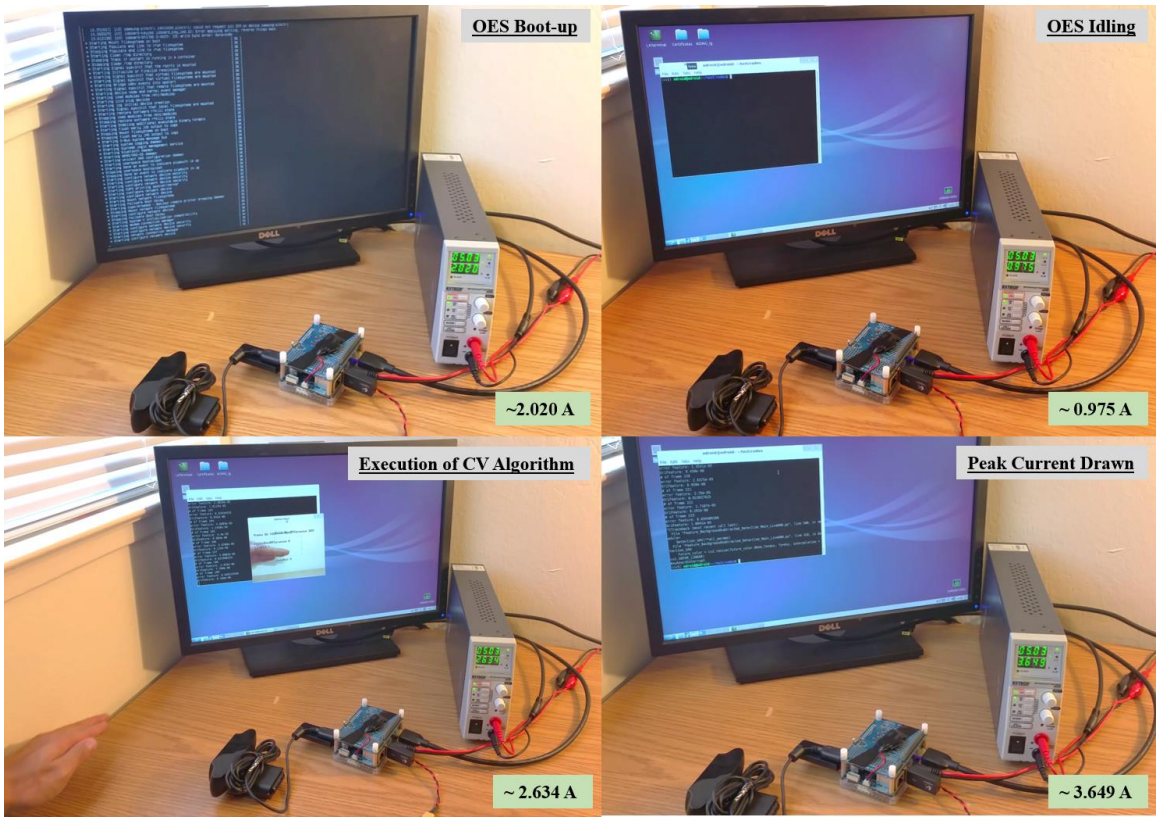


Figure 43. Current Drawn by OES during Various Phases of Operation.

b. Flight Tests

Upon completion of the ground checks, the prototype UAV subsystem underwent testing for real-time flight operation at low altitude of less than 30 feet. While the UAV was in flight, the CV algorithm was executed to detect and track multiple moving objects, which comprised passing vehicles. As shown in Figure 44, the CV algorithm was able to detect and track moving vehicles over numerous tests. In real-time flight operation, it was able to discern the moving objects from background clutter and stationary vehicles consistently.

Figure 44. Screenshots of CV Algorithm in Prototype Verification Tests.

The prototype testing was conducted numerous times to inspire confidence and build proficiency in the operator to manage the various interfaces and flight operations. The system reliability was also verified in these repeated tests. The system then underwent an integrated evaluation.

3. Evaluation of Overall System Architecture

After consistent and safe UAV flights with normal functioning of the CV algorithm, the overall system architecture was validated. The system was tested in operational conditions with varying designs as described in the following sections.

a. Experiment Setup

In addition to the system architecture of the Matrice 100 and P3at UGVs, the 3DR Solo and DJI Inspire 1 quadcopters were deployed to demonstrate the proof of concept for detecting and tracking multiple moving targets. Refer to Figure 45 for the various unmanned systems used for the experiment.

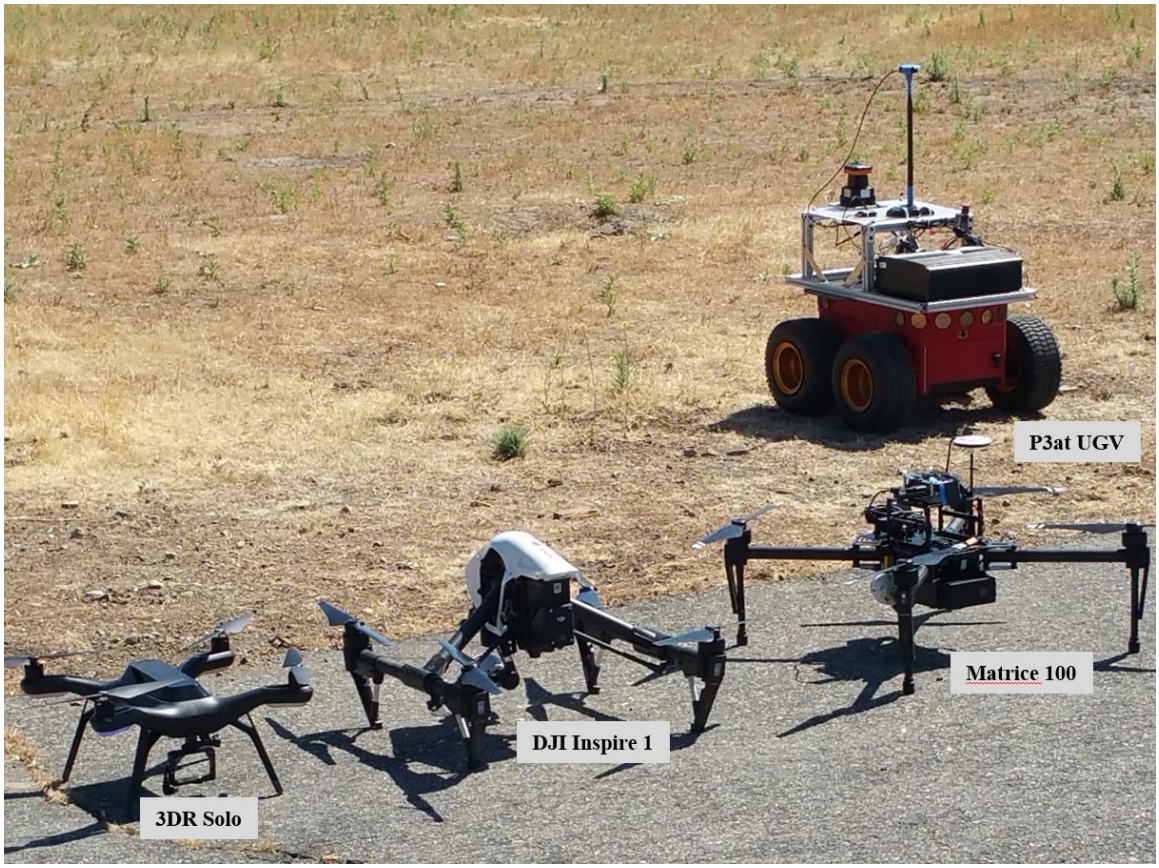


Figure 45. Various Unmanned Systems Used for Evaluation of Overall System Architecture.

The evaluation was conducted at the NPS Field Laboratory, McMillan Airfield of Camp Roberts. The large gazette area with coordinated airspace provided a safe and suitable testing ground for the experiment. The evaluation was conducted in two distinct areas within the field laboratory as demarcated in Figure 46. The concrete area spanning 30 x 20 x 30 m (L x W x H) simulated an urban environment cluttered with buildings, manned vehicles, and obstacles. The field area spanning a cuboid of 30 m in axis emulated a typical operational area with consistent and ample illumination. The overall system was tested in these distinct AOs to validate its effectiveness in varying operational environments. The height ceiling for UAV flight was capped at 30 m. The GCS was stationed in a central position to maximize the omni-directional reach of the WiFi signal for both AOs.

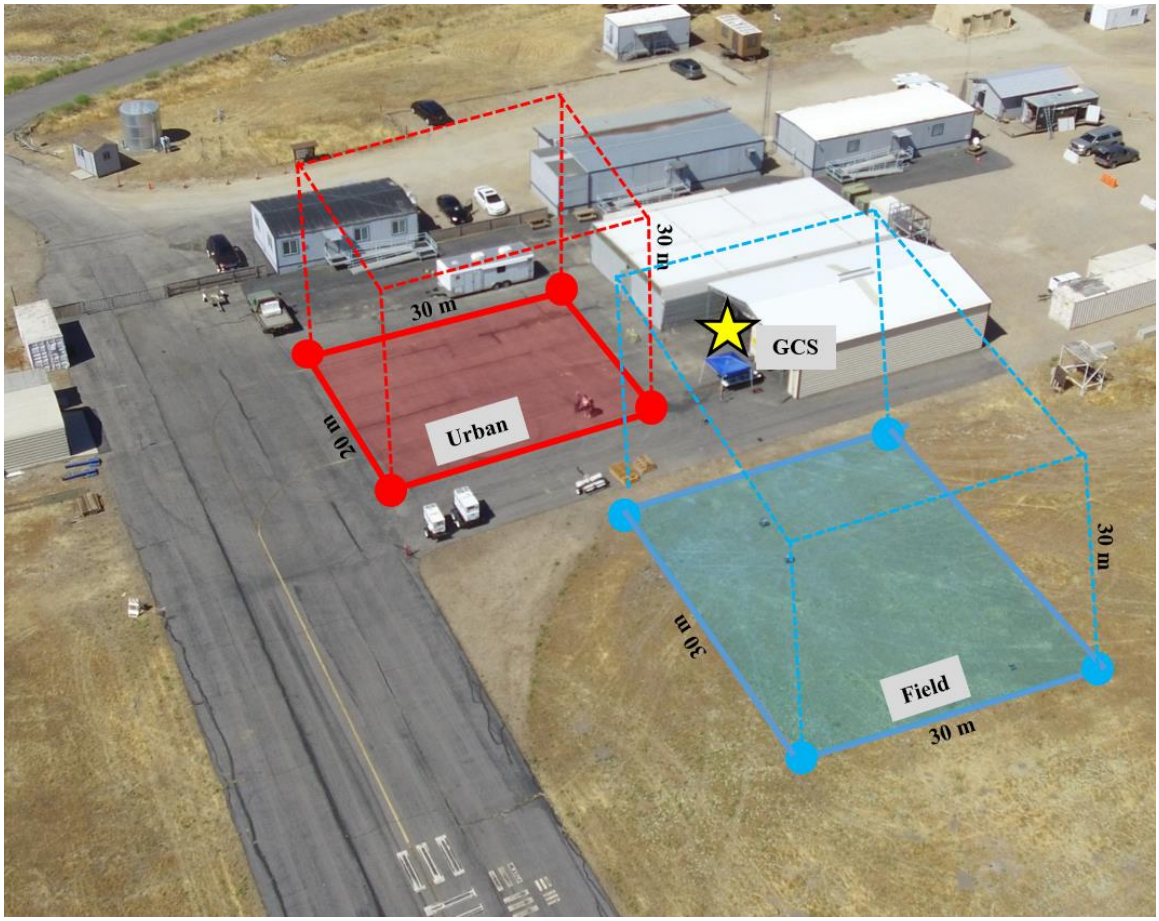


Figure 46. Two Distinct Areas of Operations in NPS Field Laboratory.

b. Experiment Results

The signal strength and the link quality of the LAN connections between the GCS and its connected devices were monitored closely. In particular, the connection between the GCS and the OES onboard the Matrice 100 was noted to determine whether the connection was sustainable for the transfer of video stream data. The Matrice 100 was flown within the confines of the boundaries, at approximately 42.42 m LOS distance away from the operator, 30 m away and 30 m high. A screenshot of the signal strength and link quality recording using Wavemon, depicted in Figure 47, shows the fluctuations over the distinct phases of Matrice 100 flight operation. The link quality maintains around -40 dB with 97% link quality when the Matrice 100 is stationary near the GCS and it deteriorates to approximately -70 dB with 54% quality during flight. These values provided empirical estimation of the link between the two devices.

Figure 47. Screenshots of Recorded Signal Strength.

The evaluation was conducted in different AOs with varying classifier parameters, as described in section A2 of this chapter. First, the evaluation was conducted in the field, with two sets of thresholds catering for small and average targets,[3] respectively. This allows the research to compare the differences in sensitivity of the CV algorithm. Second, also in field conditions, the resolution of the video feed was adjusted from 640 x 480p to 800 x 600p to observe the difference in computational efficiency. Finally, using the optimal resolution technique, the evaluation was conducted in the urban condition with the varying classifier parameters to observe the algorithm's sensitivity in a cluttered environment.

[3] The threshold for small targets uses $T_1 = 8$ and $T_2 = 0.005$, while the threshold for average targets uses $T_1 = 5$ and $T_2 = 0.02$.

(1) Field Conditions

Figure 48 shows the screenshots of the annotated videos with varying classifier parameters. The video feed was set at 640 x 480p. These screenshots do not provide comprehension of the entire footage. It was observed that setting the classifier parameters for small targets resulted in enhanced sensitivity. Comparatively, classifier parameters for average targets resulted in more misdetections as the CV algorithm was unable to discern the slower moving UGVs and smaller UAV.

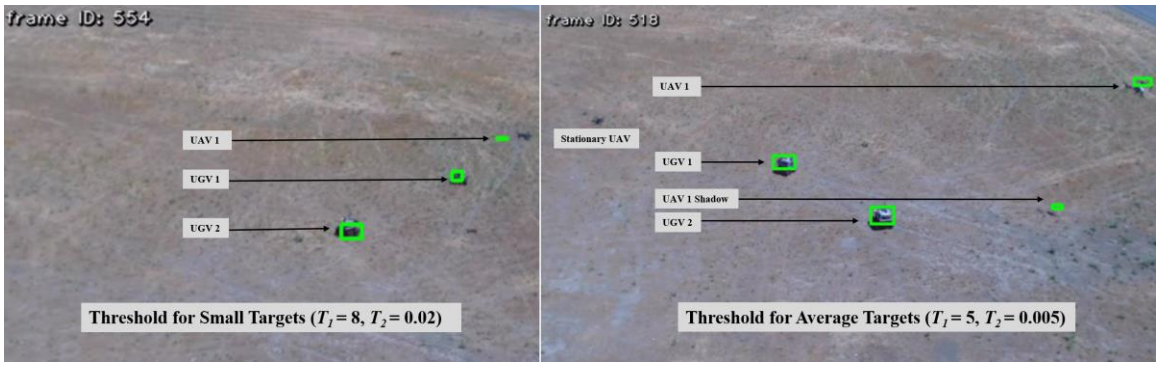


Figure 48. Varying Classifier Parameters at 640 x 480p in Field Conditions.

Next, the classifier parameters for small targets were kept while the resolution of the video feed was altered between 480p and 600p. It was observed that the sensitivity in both resolutions was similar, with the CV algorithm having some misdetections and false tracking in both scenarios. There were no observable differences in computational efficiency between the two resolutions as depicted in Figure 49. To this effect, the 600 p has a wider field of view, thereby encoding more information. It also boasts greater clarity for better recognition and identification of targets. However, 600 p with more information requires high data transfer that is limited by the type of LAN, bandwidth, and transmission range. In this evaluation, the OES is in relatively close proximity to the GCS. Thus, there was no apparent diminishing effect on the transmission of data, which would affect the latency of the annotated video feedback. It is important to consider the trade-off between the resolution of video and the latency of video feedback for operation over longer range.

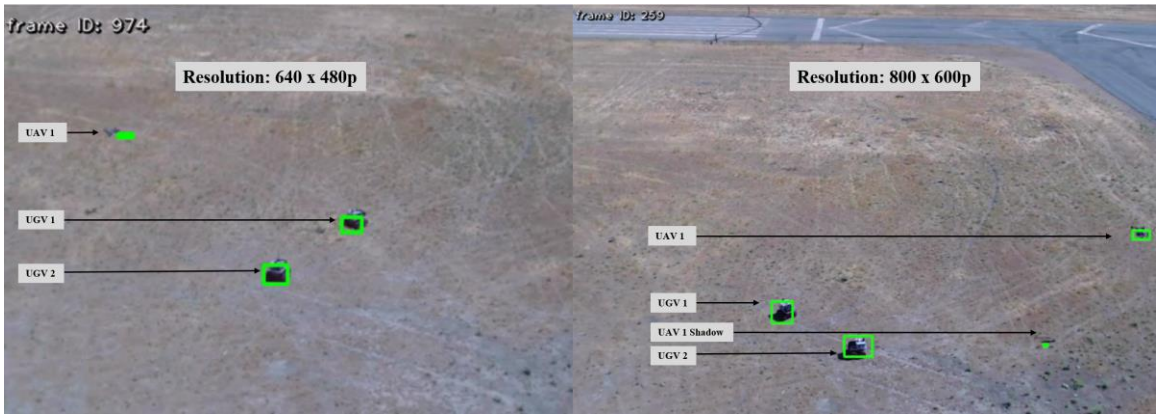


Figure 49. Varying Resolution between 480p and 600p in Field Conditions.

(2) Urban Conditions

Using the pre-determined 600p resolution from the evaluation in field conditions, the CV algorithm was executed in urban conditions, which is relatively noisier with background clutter. The evaluation was conducted with the two sets of classifier parameters. Figure 50 depicts the screenshots of the annotated videos with varying classifier parameters. In general, the cluttered AO proved to be challenging. It was observed that at 600p in urban conditions, the classifier parameters catering for small targets worked much better than parameters catering for average targets. There were many more misdetections and occurrences of faulty tracking in the latter. For example, the threshold for average targets was able to pick up the larger, faster human target but was unable to discern the slower, smaller UGV.

Figure 50. Varying Classifier Parameters at 800 x 600p in Urban Conditions.

The test was repeated using only UGVs as moving targets. It was conducted at 600p with classifier parameters set for a small target. This replication of the test verifies that the current settings for the CV algorithm were able to discern the slow and small UGVs from the background clutter. In addition, the algorithm was able to accurately detect the UGVs after they resumed movement from their stop positions, as shown in Figure 51.

Figure 51. Classifier Parameters for Small Targets at 600p in Urban Conditions.

(3) Post-experimental Testing

With the noisy background clutter and obstacles, urban conditions proved to be challenging to obtain consistent detection and tracking. Additional laboratory testing was conducted on recorded live video stream in the post experiment. The CV algorithm parameters were further tuned to enhance the sensitivity. The thresholds for clustering and grouping of points[4] were lowered to capture and maintain tracking of the minute movements of the targets in a noisy background. As shown in Figure 52, the tuned CV algorithm was able to pick up the human subject, the UGV, and the UAVs simultaneously. This promising result prompts further research in future work.

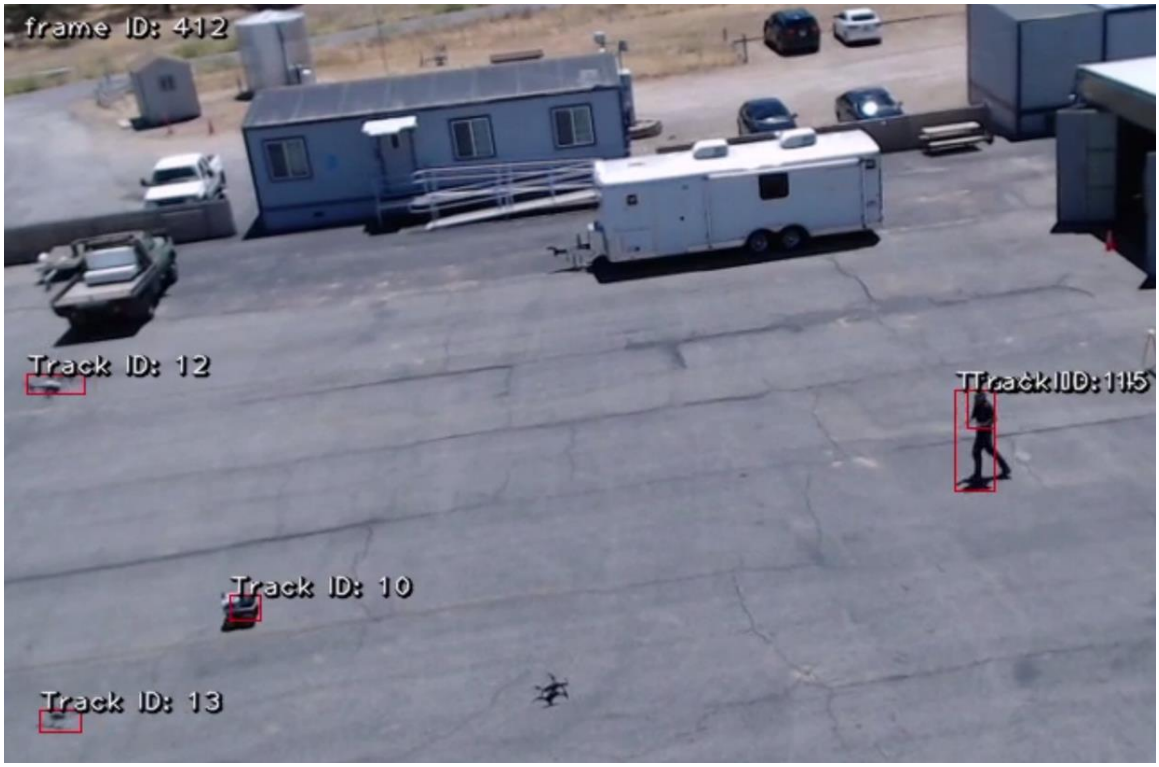


Figure 52. Post-experimental Testing with Tweaked Parameters.

[4] In the CV algorithm, the threshold for *dilate* controls the spatial proximity of the points to be determined as a cluster. The threshold for *Kernel size* governs the spatial proximity of the cluster to be grouped as a single moving object.

(4) Oversensitivity

While classifier parameters for small targets repeatedly produced consistent detection and tracking in different AOs, over-tuning or too low a threshold produces false positives and unwanted noise. The oversensitive CV algorithm will pick up unnecessary minute movements caused by UAV propeller down-wash or even strong winds, as shown in Figure 53. There is a need to fine-tune the sensitivity of the CV algorithm to capture small, slow objects while blocking out unwanted noise.

Figure 53. Unwanted Noise from Oversensitive Tuning.

(5) Areas for Improvements

At this point in this research, the CV algorithm is still in the midst of revision and updates to provide for a robust operation. The current version used in this research occasionally produces expanding bounding boxes when the detected moving targets cross

paths and overlap, as depicted in Figure 54. This reduces the effectiveness in distinguishing the individual moving targets.

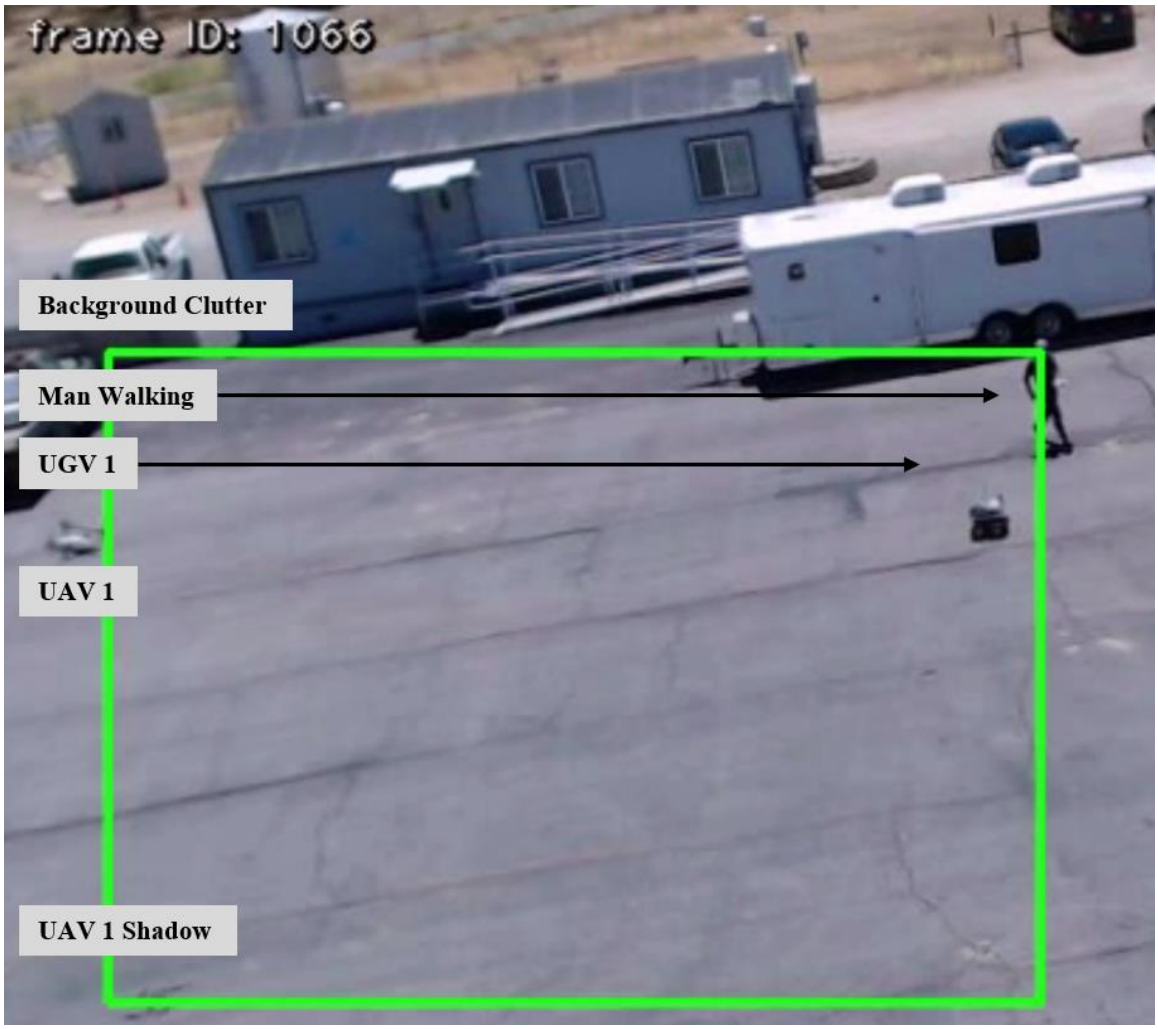


Figure 54. Expanding Bounding Boxes of Detected Targets.

Environmental conditions such as strong winds influence the flight of the UAV, which, in turn, affects the operation of the CV algorithm. Detection and tracking is less consistent in strong winds, as shown in Figure 55.

Figure 55. Operation of CV Algorithm in Strong Wind Conditions.

In addition, strong wind or fast body movements of the UAV causes rolling shutter effects due to the CMOS camera used. The rolling shutter effects, highlighted in Figure 55 and Figure 56, reduce the effectiveness of the CV algorithm in detection and tracking.

Figure 56. Comparison between Rolling Shutter and Status Quo.

C. DATASET OF ANNOTATED GROUND-TRUTH VIDEOS

Purdue University and NPS have collaborated to generate a dataset of annotated videos reflecting the ground-truth of multiple moving objects. The intent is to establish a comprehensive public reference source for future CV or motion related researches, spurring advancements in respective fields. By comparing against research works, these ground-truth annotated videos can be used for performance evaluation. At the time of this

writing, the collaboration effort has produced 50 annotated videos that are released publicly by Bouman[5] (2017) with more videos in the midst of annotation.

These point-of-view videos are recorded either by a GoPro 3 camera mounted on a custom delta-wing airframe or by a Zemuse X3 gimbal camera on a DJI Matrice 100. All videos are of HD resolution at 1920 x 1080p. There are moving objects in each video that vary in appearance, size, and shape. These moving objects include UGVs, quadcopters, fixed-wing UAVs, humans, and vehicles.

The dataset is manually annotated frame-by-frame by both institutions using the open platform Video Annotation Tool from Irvine California (VATIC) software. VATIC is a public framework that is extensively used in the crowdsource market for video-labelling projects. It has an intuitive video annotation user interface designed to reduce the cognitive load of the annotator (Vondrick, Patterson, and Ramanan 2012). Figure 57 shows a snippet of the manual annotation process using VATIC software.

Figure 57. Manual Annotation of Ground-Truth Video Using VATIC.

[5] The dataset of annotated videos is published online at https://engineering.purdue.edu/~bouman/UAV_Dataset.

V. CONCLUSION AND RECOMMENDATIONS

A. SUMMARY

This thesis integrated and assessed the use of EO sensors to achieve detect and track capability in multi-UAV operations. A systems engineering approach was adopted to facilitate the design and production of the overall system architecture. It defined the issue at hand, analyzed the stakeholder needs and limitations, decomposed the mission and functional requirements, synthesized a proposed system architecture, and proceeded with the testing and evaluation of the overall system framework.

The system architecture comprised the CV algorithm executed by the OES onboard the main UAV, with multiple UGVs simulating moving targets. The GCS connected all the devices via the LAN connection, allowing operator command and control. The CV algorithm consisted of a series of sequential processes that systematically isolated the moving objects by background subtraction, pruning and classifying them as targets, before commencing tracking. The UGVs were programmed for autonomous waypoint navigation that allowed for minimal operator interfaces.

The overall system was tested and evaluation was performed in progressive stages, ranging from components and subsystems testing to overall system evaluation. It was observed that a higher resolution video feed provides a wider field of view and better clarity, but this is at the expense of additional data requirements that may result in increased latency of the video feedback. The classifier parameters were tuned to capture both small and average targets. While tuning for small targets produced a more sensitive and consistent detection and tracking capability, caution must be exercised to prevent over-tuning. Over-tuned parameters result in the accommodation of unwanted noise. Furthermore, it was observed that the developing CV algorithm faced some artifacts such as expanding bounding boxes and reduced effectiveness in strong wind and rolling shutter effects.

In conclusion, this thesis has demonstrated that the use of EO sensors onboard a functional UAV to detect and track multiple moving targets is feasible. By doing so, it

can greatly reduce an operator's cognitive load and enhances the operator's situation awareness in an otherwise crowded environment. As also demonstrated, the use of a lightweight, existing onboard camera minimizes the UAV carrying load, allowing for better maneuverability and longer endurance than traditional sensors such as LIDAR.

B. RECOMMENDATIONS FOR FUTURE WORK

Future work is recommended to improve the current system architecture. From the hardware perspective, magnetic shielding can be applied to the OES, allowing it to be installed closer to the UAV frame. This should improve structural integrity and streamline the overall profile of the UAV. Such a change will also enhance the aerodynamics by reducing the drag forces. In addition, a more powerful onboard processor such as the NVIDIA Jetson TX2 can be used in place of the Odroid-XU4. With the boosted processing speed, more computationally intensive functions such as real-time HD video processing can be executed efficiently.

Tests can be performed on the CCD camera sensor instead of the C920 CMOS sensor to observe the effects of global versus rolling shutter, respectively. This may allow the CV algorithm to operate effectively in strong wind or fast body movement conditions. Alternatively, future research could tap into the organic Zenmuse X3 camera. The gimbal camera provides stable HD video stream despite external motion influences, and it eliminates the need for an additional camera.

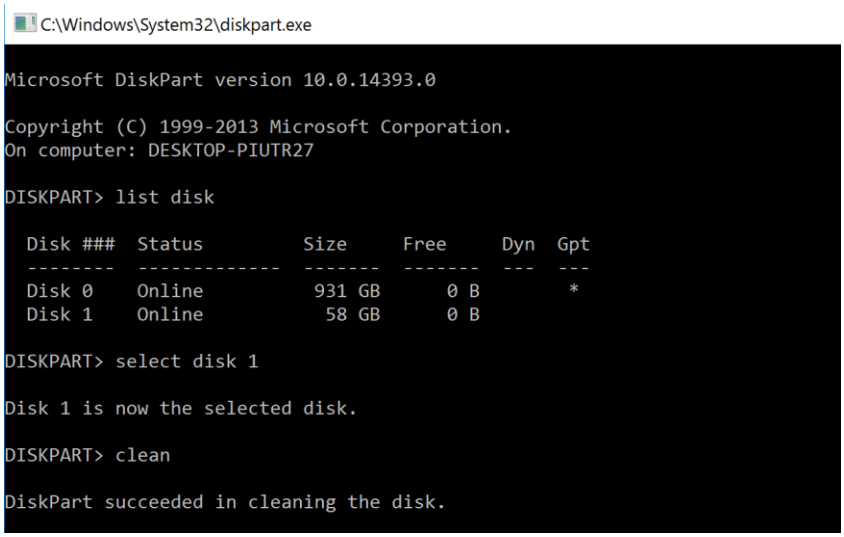
From the software perspective, more experiments can be conducted in the tuning of the parameters, such as tweaking of the motion difference threshold, clustering size, and Kalman filter tracking. This will allow fine-tuning of the CV algorithm specifications for different real-time operational scenarios. The CV algorithm can be integrated into the ROS architecture and the DJI onboard SDK. By translating the CV algorithm bounding box coordinates outputs into the ROS topic, the DJI onboard SDK will be able to harness these ROS outputs to control the Matrice 100 UAV position. Future work can then implement vision-based attacking or avoiding algorithms based on the CV algorithm and this ROS architecture.

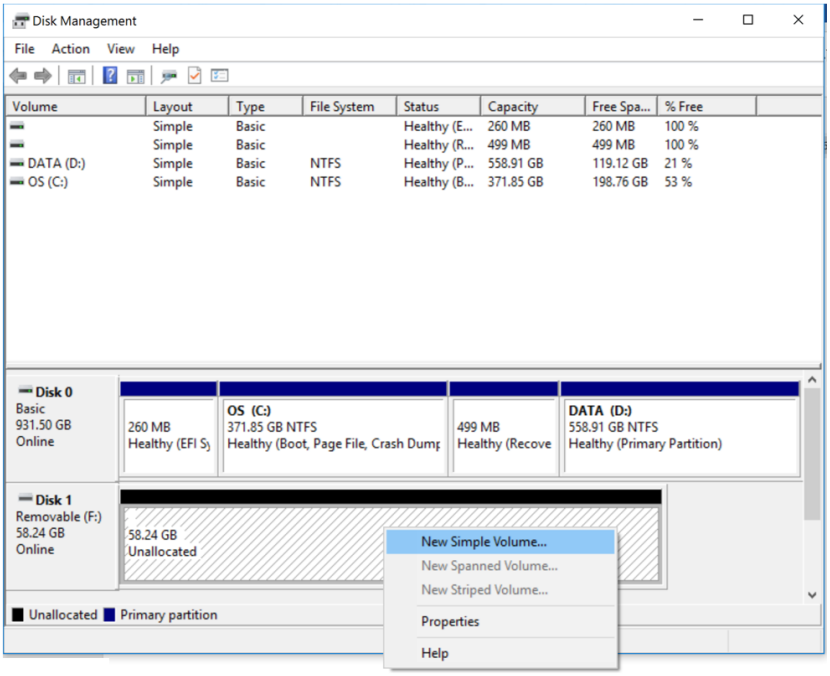
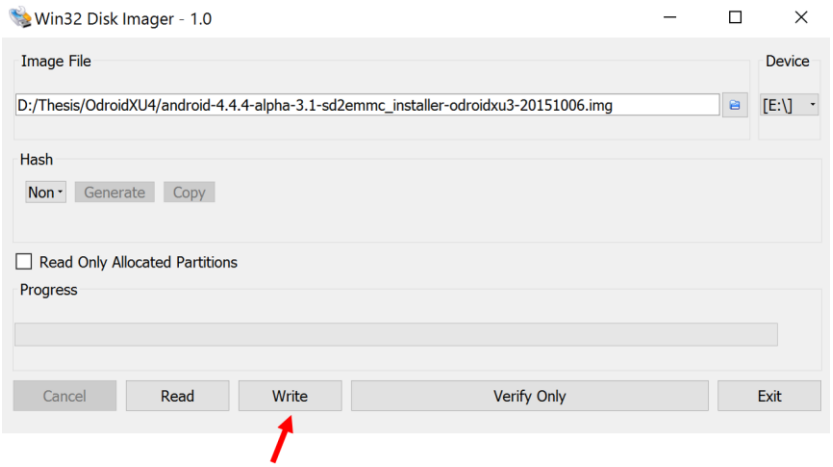
APPENDIX A. SETUP OF THE ODROID-XU4

The Odroid-XU4 has to be setup and flashed with the relevant software prior to any downstream research work. The preparation process includes the setup of the storage card, flashing of the Android 4.4.4 Alpha 4.5 and Ubuntu 14.04 to the OES itself. The following sections list the instructional steps for each process. The procedures are sourced and adapted from the Odroid Forum (2017).

A. PREPARATION OF THE STORAGE CARD (WINDOWS)

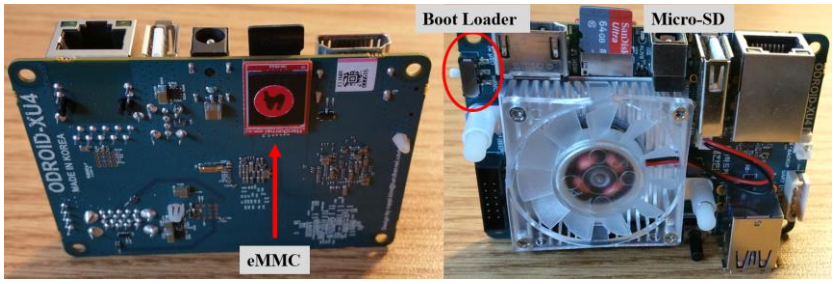
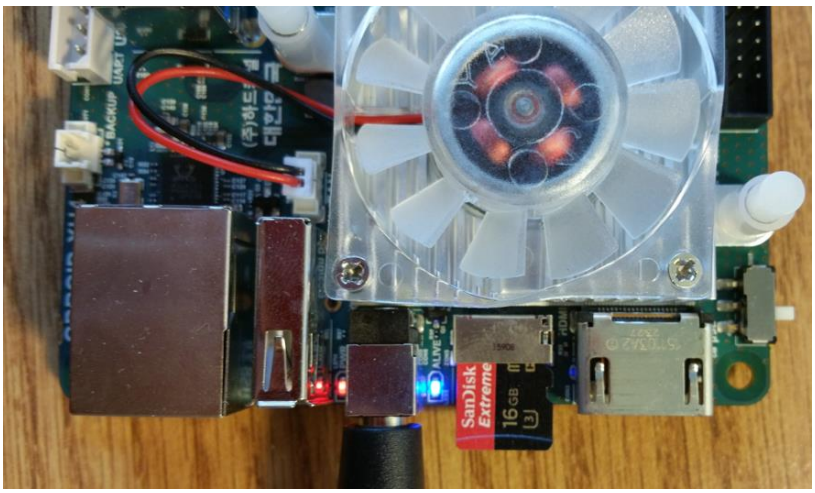
The storage devices are prepared and flashed with the necessary software, namely the Android and Ubuntu operating systems. This software will subsequently be installed into the OES.

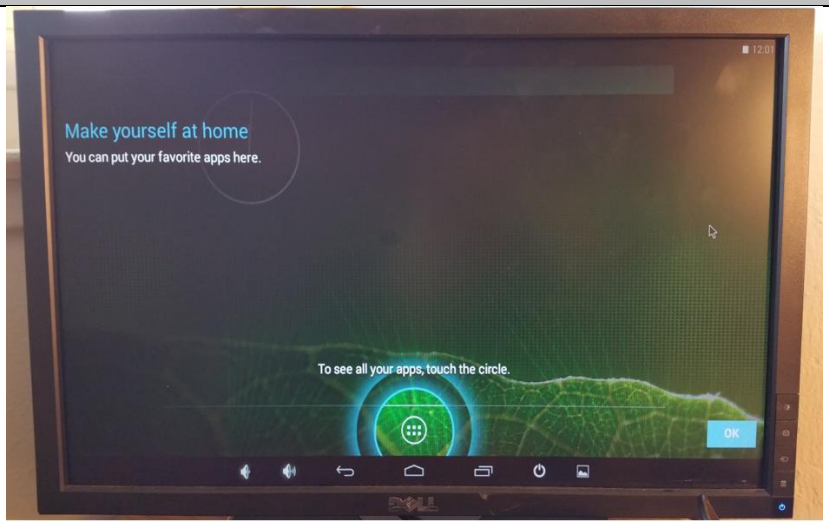
Commands	**Comments**
	Connect eMMC or micro-SD card to the computer via an adaptor.
Type in *diskpart*	Search in Windows for diskpart. This pulls up a command terminal.
Type in *list disk*	See which disk # is in the storage card.
Type in *select disk #*	Select the disk # with the storage card.
Type in *clean #*	Clean the card. 
	- Right click on Windows button on task bar. - Look for **Disk Management.**

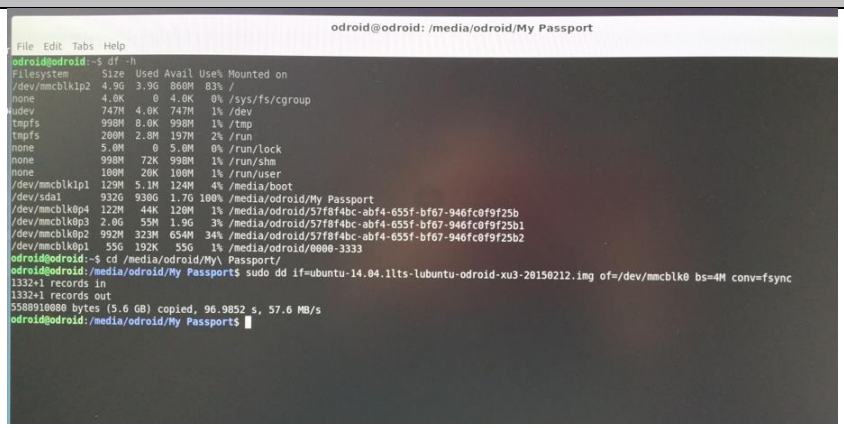
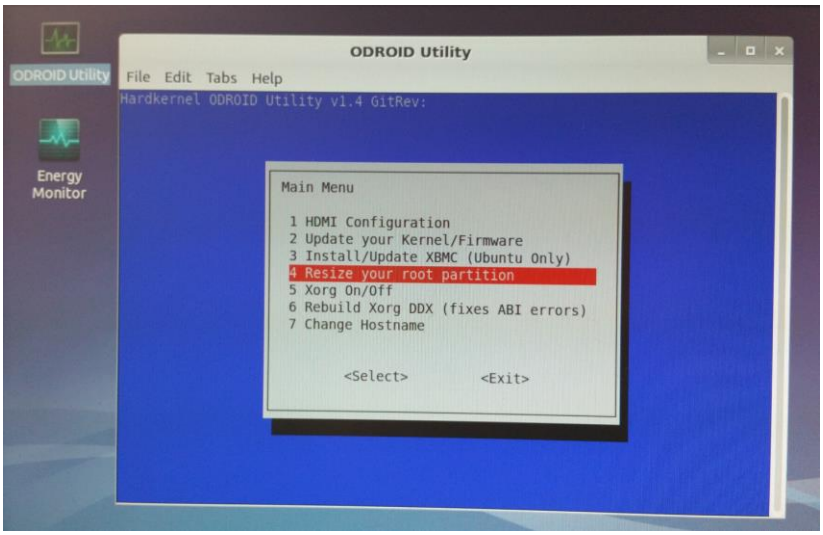
Commands	Comments
	- Right Click Disk 1 > New Simple Volume > Format 
	- Download and uncompress the Android image file. *<android-4.4.4-alpha-4.5-sd2emmc_installer-odroidxu3-20170315.img.zip>* - Install image writer software **Wind32disk** imager. - Flash the uncompressed Android image file onto the micro-SD card 1. 
	- Download and uncompress the Ubuntu image file. *< ubuntu-14.04.1lts-lubuntu-odroid-xu3-20150212.img >* - Flash the uncompressed Ubuntu image file onto the micro-SD card 2. - Copy the Ubuntu image file onto a USB stick.

B. FLASHING THE OPERATING SYSTEM INTO OES

This section describes the flashing of the Android and the Ubuntu operating systems loaded in the storage card into the OES.

Commands	Comments
	- Connect both eMMC and micro-SD card 1 to the Odroid-XU4. - Ensure the Odroid-XU4 is set to boot on SD jumper, not eMMC jumper. 
	- Boot up the Odroid-XU4. - Look for a slow flashing blue LED light and a solid red light. - Wait for 15 minutes to complete flashing process. - Power off the Odroid-XU4. 
	- Remove the micro-SD card, leaving eMMC connected. - Set the Odroid-XU4 to eMMC jumper. - Boot up the Odroid-XU4 to check if Android OS is flashed properly.

Commands	Comments
	
In command terminal Type *ls/dev/mmcblk**	- Connect the eMMC, micro-SD card 2, and USB stick to the Odroid-XU4. - Ensure the Odroid-XU4 is set to boot on SD jumper, not eMMC jumper. - Boot up the Odroid-XU4. - Check location of eMMC card, which will be the device with boot0/boot1/p1/p2/p3. Usually, it is mmcblk0. 
- Type *df –h* - *cd /media/odroid/name* - *sudo dd if =Ubuntu-14.04.1lts-ubuntu-odroid-xu3-20150212.img of=/dev/mmcblk bs=4M conv=fsync*	- Navigate the USB stick file directory where the Ubuntu image in stored. - Flash the Ubuntu image onto the Odroid-XU4. Default password required is odroid.

Commands	Comments
	
	- Power off, remove micro-SD, and USB stick. - Set to eMMC jumper and boot up. - Run Odroid Utility, resize partition to ensure proper allocation of hard disk. 

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. INSTALLATION OF PYTHON AND OPENCV

After flashing the Android and Ubuntu OS, the author installed the various software used for the thesis research; namely, the Python 2.7, OpenCV 3.0, Linux Screen, Wavemon, and Simple Screen Recorder. This appendix describes the installation of Python 2.7 and OpenCV 3.0. The procedures are sourced and adapted from Py Image Search (2017). Prior to commencing installation, ensure that the Odroid-XU4 has a stable Internet connection with an attached WiFi adapter.

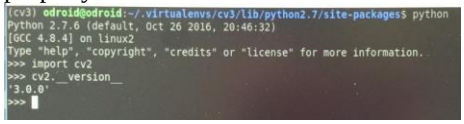
A. PREPARATION OF THE ODROID-XU4 FOR INSTALLATION

This section describes the procedures for preparing the OES for downstream configuration. It involves the configuration of the system libraries, firmware upgrades, and updates.

Commands	**Comments**
- *sudo apt-get update* - *sudo apt-get upgrade*	- Update and upgrade the Ubuntu OS.
- *sudo apt-get install gedit* - *sudo apt-get install build-essential git cmake pkg-config* - *sudo apt-get install libjpeg8-dev libtiff5-dev libjasper-dev libpng12-dev* - *sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev* - *sudo apt-get install libgtk2.0-dev* - *sudo apt-get install libatlas-base-dev gfortran*	- Install the note editor. - Install the required developer tools, image and video I/O packages, GTK development library, and optimization packages.
- *cd ~* - *git clone https://github.com/Itseez/opencv.git* - *cd opencv* - *git checkout 3.0.0* - *cd ~* - *git clone https://github.com/Itseez/opencv_contrib.git* - *cd opencv_contrib* - *git checkout 3.0.0*	- Retrieve the OpenCV repository from the community folder in Github. * git –c http.sslVerify=false clone https://github.com/Itseez/opencv.git# for proxy requiring ssl certification

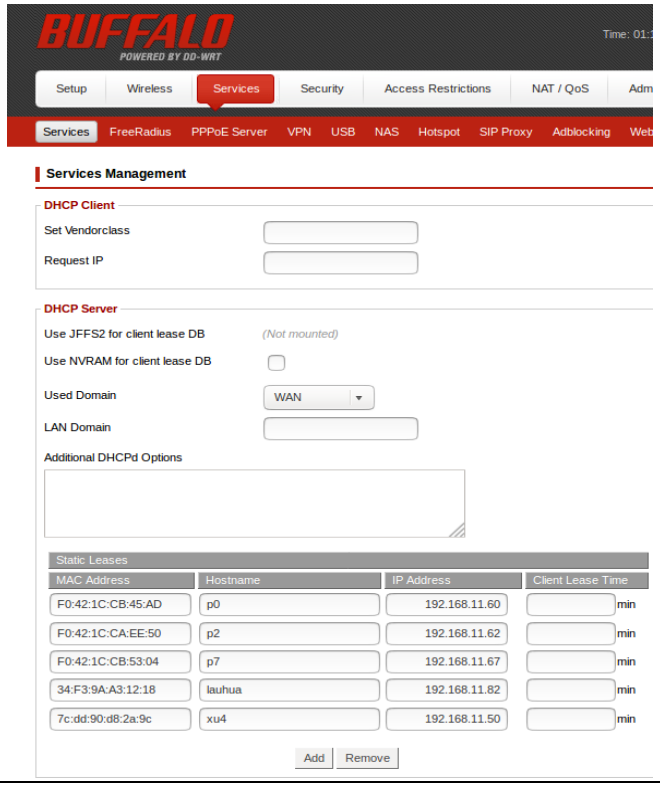
B. COMPILING PYTHON AND OPENCV

This section describes the procedures for installing the software Python 2.7 and OpenCV 3.0 used in the CV algorithm.

Commands	Comments
- *sudo apt-get update* - *sudo apt-get upgrade* - *sudo apt-get install python2.7-dev* - *wget https://bootstrap.pypa.io/get-pip.py* - *sudo python get-pip.py* - *sudo pip install virtualenv virtualenvwrapper* - *sudo rm -rf ~/.cache/pip* - *gedit ~/.bashrc* - *source ~/.bashrc*	- Install Python 2.7 header files. - Install pip, python package manager. - Install virtualenv and virtualenvwrapper. - Update the bashrc file to automatically source for the files by adding these lines to the bottom of the file: # virtualenv and virtualenvwrapper export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python2.7 export WORKON_HOME=$HOME/.virtualenvs source /usr/local/bin/virtualenvwrapper.sh
- *mkvirtualenv cv3* - *workon cv3* - *pip install numpy*	- Create a virtual environment for the CV operation. - Install Numpy inside the cv3 virtual environment.
- *cd ~/opencv* - *mkdir build* - *cd build* - *cmake -D CMAKE_BUILD_TYPE=RELEASE * *-D CMAKE_INSTALL_PREFIX=/usr/local * *-D INSTALL_C_EXAMPLES=ON * *-D INSTALL_PYTHON_EXAMPLES=ON * *-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules * *-D BUILD_EXAMPLES=ON * *-D WITH_OPENMP=ON ..* - *make -j8* - *sudo make install* - *sudo ldonfig*	- Build the cv3 setup. - Compile cv3. - Install cv3.
- *cd ~/.virtualenvs/cv3/lib/python2.7/site-packages/* - *ln -s /usr/local/lib/python2.7/site-packages/cv2.so cv2.so* 	- Create a hard link in the site-packages directory of the cv3 environment. - Verify that Python and OpenCV3 are installed properly. 

APPENDIX C. ESTABLISHING A LAN CONNECTION

A Buffalo N600 DD-WRT router was used to establish and bridge LAN connections among the connected devices such as the OES and multiple UGVs. This appendix describes the procedures for creating the LAN and the subsequent connection process.

Commands	**Comments**
- *ifconfig -a*	- Identify and note the various IP and MAC addresses of the client devices to be connected via the LAN.
	- Boot up the N600. Connect a computer device to the router network using the SSID and password. - Access the N600 router configuration page via 192.168.11.1 using the default username and password. - Under the Services tab, add the static DHCP address configuration. 
- *xhost +* - *ssh odroid@xu4 -X*	- Connect all devices under the established LAN. - Access the devices via their assigned hostname.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. SETUP AND CONTROL OF UGVs

The UGVs are connected to the established LAN of the GCS. The procedures for setting up and controlling the UGVs were sourced and adapted from the NPS MRC Wiki page, created by Bingham (2017).

A. SETUP OF UGVs

This section describes the installation of necessary software and the configuration to prepare the P3ats to receive both manual and autonomous navigation commands.

Commands	**Comments**
- *sudo apt-get install ros-indigo-multimaster-fkie*	- Install the multi-master ROS package.
- *sudo gedit /etc/sysctl.conf* - *sudo service procps restart* - *cat /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts*	- Set up laptop for multicast networking. - Add this line to the end of the sysctl file: net.ipv4.icmp_echo_ignore_broadcasts=0 - Verify that multicast is enabled.
- *cd ~/catkin_ws/src* - *git config http.sslverify false* - *git clone https://github.com/bsb808/nre_multimaster.git* - *cd ~/catkin_ws* - *catkin_make* - *source devel/setup.bash* - *sudo gedit /src/nre_multimaster/launch/sync.launch*	- Set up the multi-master_fkie package - Add these lines to the bottom of the sync.launch file: <node name="master_sync" pkg="master_sync_fkie" type="master_sync" output="screen"> <!-- Ignore all hosts by default --> <rosparam param="ignore_hosts">['*']</rosparam> <!-- Add selective hosts to sync with --> <rosparam param="sync_hosts">[192.168.11.60,192.168.11.61]</rosparam> </node>
- *cd ~/catkin_ws/src* - *git config http.sslverify false* - *git clone https://github.com/bsb808/nre_joy.git* - *cd ~/catkin_ws* - *catkin_make* - *source devel/setup.bash*	- Set up the joystick control ROS package

B. CONTROL OF UGVs

This section lists the various control commands used to boot up and operate the P3ats. The autonomous waypoint navigation scripts are not listed here.

Commands	Comments
- *ifconfig #* see which wlan the device is on. - *sudo route add –net 224.0.0.0 netmask 224.0.0.0 wlan1* - *roslauch nre_multimaster sync.launch*	- Ensure that the UGVs are connected to the LAN. - Launch the nre_multimaster sync file.
- *roslaunch nre_joy teleop_ugv_joy_p3at.launch namespace:=p0 autorepeat_rate:=10*	- Launch and control via joystick the specific P3at (e.g.) p0.
- *ssh frl@p0* - *sudo service nre restart* - *rosnode list* - *rostopic echo /p0/nav/status*	- Ensure that the P3at status is ready.

LIST OF REFERENCES

Ang, Kelvin. 2016. "Vision-Aided Navigation." Lecture, Advanced Robotics Center, National University, Singapore, Singapore, July 22, 2016.

Aqel, Mohammad O. A., Mohammad H. Marhaban, M. Iqbal Saripan, and Napsiah Bt. Ismail. 2016. "Review of Visual Odometry: Types, Approaches, Challenges, and Applications." *SpringerPlus* 5(1): 1897. doi:10.1186/s40064-016-3573-7.

Blanchard, Benjamin, and Wolter Fabrycky. 2010. *Systems Engineering and Analysis*. 5th ed. Englewood Cliffs, NJ: Prentice-Hall.

Bosch, Anna, Andrew Zisserman, Xavier Mu, and Xavier Munoz. 2007. "Image Classification Using Random Forests and Ferns." *IEEE 11th International Conference on Computer Vision (ICCV)*: 1–8. doi:10.1109/ICCV.2007.4409066.

Bouman, Charles. 2017. "UAV Dataset." Accessed June 14. https://engineering.purdue.edu/~bouman/UAV_Dataset/.

Bingham, Brian. 2017. "NPS Wiki Multi Robot Control." Accessed June 14, 2017. https://wiki.nps.edu/display/MRC/Multi+Robot+Control+Home.

Brown, Timothy X., Sheetalkumar Doshi, Sushant Jadhav, and Jesse Himmelstein. 2004. "Test Bed for a Wireless Network on Small UAVs." *AIAA Third Unmanned Unlimited Technical Conference*, Chicago, IL: 20–23.

Brox, Thomas, and Jitendra Malik. 2011. "Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation." *IEEE Transactions on Pattern Analysis and Machine Intelligence Vol.* 33: 500–513. doi:10.1109/TPAMI.2010.143.

Buffalo Americas, Inc. 2017. "N600 DD-WRT Wireless Router." Accessed July 21. http://www.buffalotech.com/products/airstation-highpower-n600-gigabit-dual-band-open-source-dd-wrt-wireless-routers.

Buede, Dennis M., and William D. Miller. 2016. *The Engineering Design of Systems: Models and Methods*. 3rd ed. Berkeley Heights, NJ: John Wiley & Sons.

Canis, Bill. 2015. *Unmanned Aircraft Systems (UAS): Commercial Outlook for a New Industry*. CRS Report No. 44192. Washington, DC: Congressional Research Service. https://fas,org/sgp/crs/misc/R44192.pdf.

Castle Creations. 2017. "Castle Creations BEC Switching Regulator." Accessed August 19. http://www.castlecreations.com/en/cc-bec-pro-010-0004-01.

Collins, Robert. 2007. “Lecture 30 : Video Tracking : Lucas-Kanade,” *Computer* C (30), CSE486. The Pennsylvania State University. http://www.cse.psu.edu/~rtc12/CSE486/lecture30.pdf.

Cohn, Jeffrey F., Adena J. Zlochower, James J. Lien, and Takeo Kanade. 1998. “Feature-Point Tracking by Optical Flow Discriminates Subtle Differences in Facial Expression.” In Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition: Technical Communication Services 396–401.

Cui, Jin Q., Shupeng Lai, Xiangxu Dong, and Ben M. Chen. 2015. “Autonomous Navigation of UAV in Foliage Environment.” *Journal of Intelligent & Robotic Systems* 84*:* 259–276. doi:10.1007/10846-015-0292-1.

Defense Acquisition University. 2000. “Introduction to Systems Engineering.” In *Defense Acquisition Guidebook (DAG)*, 1–109. https://www.dau.mil/tools/dag/Pages/DAG-Page-Viewer.aspx?source=https://www.dau.mil/guidebooks/Shared%20Documents%20HTML/Chapter%203%20Systems%20Engineering.aspx

DJI Drones. 2017. “Matrice 100 UAV Specifications.” Accessed July 14. https://www.dji.com/matrice100.

Federal Aviation Administration. 2017. “Unmanned Aircraft Systems.” Last modified July 3. https://www.faa.gov/uas/.

Fierro, Rafael, Aveek Das, John Spletzer, Joel Esposito, Vijay Kumar, James P. Ostrowski, and George Pappas. 2002. “A Framework and Architecture for Multi-robot Coordination.” *International Journal of Robotics Research* 21: 977–995. doi:10.1007/3-540-45118-8-31.

Gonzalez, Ramon, Francisco Rodriguez, Jose Luis Guzman, Cedric Pradalier, and Roland Siegwart. 2012. “Combined Visual Odometry and Visual Compass for off-Road Mobile Robots Localization.” *Robotica* 30(6): 865–78. doi:10.1017/S026357471100110X.

Hardkernel. 2017. “Instructions for Odroid XU4 Setup.” Accessed May 4. https://forum.odroid.com/.

Han, Sekyung, and Jawhwan Lim. 2013. “Performance Assessment of a Lithium-Polymer Battery for HEV Utilizing Pack-Level Battery Hardware-in-the-Loop-Simulation System” *Journal of Electrical Engineering and Technology* 8:1431–38.

Hilliard, Rich. 2012. “Architecture Viewpoint Template.” *ISO/IEC/IEEE 42010:2011*.

Horn, Berthold K.P., and Brian G. Schunck. 1981. “Determining Optical Flow.” *Artificial Intelligence* 17: 185–203.

Jenkins, Darryl, and Bijan Vasigh. 2013. *The Economic Impact of Unmanned Aircraft Systems Integration in the United States*. Arlington, VA: Association for Unmanned Vehicle Systems International. http://www.auvsi.org/econreport.

Kon, Tayfun. 1998. "Collision Warning and Avoidance System for Crest Vertical Curves." Master's thesis, Virginia Tech, Blacksburg, VA.

Lee, Jae-Neung, and Keun-Chang Kwak. 2014. "A Trends Analysis of Image Processing in Unmanned Aerial Vehicle." *International Journal of Computer, Information Science and Engineering* 8(2): 261–64.

Lenovo. 2017. "Thinkpad T460." Accessed July 21. htttp://www3.lenovo.com/us/ /thinkpad/thinkpad-t-series/T460/p/22TP2TT4600.

Jing Li, Dong Hye Ye, Timothy Chung, Mathias Kolsch, Juan Wachs, and Charles Bouman. 2016. "Multi-Target Detection and Tracking from a Single Camera in Unmanned Aerial Vehicles (UAVs)." *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems:* 4992–97. New York: Institute of Electrical Electronics Engineers. doi:10.1109/IROS.2016.7759733.

Liu, Ce. 2011. "Motion Estimation." Lecture, Computer Science Department, University of Washington, Seattle, September 5.

Logitech. 2017. "HD Pro Webcam C920." Accessed August 19. https:// www.logitech.com/en-us/product/hd-pro-webcam-c920.

Lønmo, Linda, and Muller Gerrit. 2014. "Concept Selection – Applying Pugh Matrices in the Subsea Processing Domain." *INCOSE International Symposium*, 24: 583–598.

Lucas, Bruce D., and Takeo Kanade. 1981. "An Iterative Image Registration Technique with an Application to Stereo Vision." *International Joint Conference on Artificial Intelligence* 7: 674–679. doi:10.1145/358669.358692.

Mathur, Vivek Narain, Andrew D.F. Price, Simon Austin, and Cletus Moobela. 2007. "Defining, Identifying and Mapping Stakeholders in the Assessment of Urban Sustainability." *2007 International Conference on Whole Life Urban Sustainability and Its Assessment*: 18. https://dspace.lboro.ac.uk/xmlui/ handle/2134/5202.

Oliver, Nuria, Barbara Rosario, and Alex Pentland. 1999. "A Bayesian Computer Vision System for Modeling Human Interactions." *Lecture Notes in Computer Science*, 255–72. doi:10.1007/3-540-49256-9_16.

Opromolla, Roberto, Giancarmine Fasano, Giancarlo Rufino, Michele Grassi, and Al Savvaris. 2016. "LIDAR-Inertial Integration for UAV Localization and Mapping in Complex Environments." *2016 International Conference on Unmanned Aircraft Systems:* 649–56. doi:10.1109/ICUAS.2016.7502580.

Parker, Lynne. 1993. “Designing Control Laws for Cooperative Agents.” *1993 IEEE International Conference on Robotics and Automation*: 582–87. doi:10.1109/ROBOT.1993.291842.

Phang, Swee King. 2016. “Mathematical Modeling and Control of Multi-Rotor UAV.” Lecture, Advanced Robotics Center, National University of Singapore, Singapore, August 02, 2016.

Py Image Search. 2017. “Installing OpenCV3 for Python2.7.” Accessed May 6. http://www.pyimagesearch.com/2015/07/27/installing-opencv-3-0-for-both-python-2-7-and-python-3-on-your-raspberry-pi-2/.

Samet, Hanan, and Markku Tamminen. 1988. "Efficient Component Labeling of Images of Arbitrary Dimension Represented by Linear Bintrees." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10: 579-586. doi:10.1109/34.3918.

Scaramuzza, Davide, and Fraundorfer Friedrich. 2011. “Visual Odometry: Part I - The First 30 Years and Fundamentals.” *IEEE Robotics and Automation Magazine* 18: 80 – 92. doi: 10.1109/MRA.2011.943233

Serre, Thomas, Lior Wolf, Stanley Bileschi, Maximilian Riesenhuber, and Tomaso Poggio. 2007. “Robust Object Recognition with Cortex-like Mechanisms.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29 (3): 411–26. doi:10.1109/TPAMI.2007.56.

Shi, Jianbo, and Carlo Tomasi. 1994. “Good Features to Track.” *1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*: 593–600. doi:10.1109/CVPR.1994.323794.

Skolnick, Fred R., and Phillip G. Wilkins. 2000. “Laying the Foundation for Successful Systems Engineering.” *Johns Hopkins APL Technical Digest* 21 (2): 208–16.

Teo, Harn Chin. 2013. “Closing the Gap between Research and Field Applications for Multi-UAV Cooperative Missions.” Master’s thesis, Naval Postgraduate School, Monterey, CA.

Teo, Kenny. 2016. “Enhancing Autonomy of Aerial Systems via Integration of Visual Sensors into Their Avionics Suite.” Master’s thesis, Naval Postgraduate School, Monterey, CA.

U.S. Department of Defense. 2005. *Unmanned Aircraft Systems Roadmap 2005–2030*. Washington, DC: Office of the Secretary of Defense.

U.S. Department of Transportation. 2013. “Unmanned Aircraft System (UAS) Service Demand 2015–2035.” *Unmanned Aircraft System (UAS) Service Demand 2015 - 2035 - Literature Review & Projections of Future Usage*: 151. doi:DOT-VNTSC-DOD-13-01.

Vondrick, Carl, Donald Patterson, and Deva Ramanan. 2012. “Efficiently Scaling up Crowdsourced Video Annotation.” *International Journal of Computer Vision* 101(1): 184–204. doi:10.1007/s11263-012-0564-1.

Welch, Greg, and Gary Bishop. 2006. “An Introduction to the Kalman Filter.” *Practice of SIGGRAPH* 8: 1–16. doi:10.1.1.117.6808.

Wong, Kam Cheong. 2011. “Using an Ishikawa Diagram as a Tool to Assist Memory and Retrieval of Relevant Medical Cases from the Medical Literature.” *Journal of Medical Case Reports* 5(1): 120. doi:10.1186/1752-1947-5-120.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California